On the equivalence of optimal recommendation sets and myopically optimal query sets

Paolo Viappiani, Craig Boutilier

On the Equivalence of Optimal Recommendation Sets and Myopically Optimal Query Sets[☆]

Paolo Viappiani[a,1,*], Craig Boutilier[b,2]

[a] *Sorbonne Université and CNRS, UMR 7606, LIP6, F-75005 Paris, France.*
[b] *Google Research, Mountain View, California, 94043, USA.*

Abstract

Preference elicitation is an important component in many AI applications, including decision support and recommender systems. Such systems must assess user preferences, based on interactions with their users, and make recommendations using (possibly incomplete and imprecise) beliefs about those preferences. Mechanisms for explicit *preference elicitation*—asking users to answer direct queries about their preferences—can be of great value; but due to the cognitive and time cost imposed on users, it is important to minimize the number of queries by asking those that have high (expected) value of information.

An alternative approach is to simply make recommendations and have users provide feedback (e.g., accept a recommendation or critique it in some way) and use this more indirect feedback to gradually improve the quality of the recommendations. Due to inherent uncertainty about a user's true preferences, often a *set of recommendations* is presented to the user at each stage. Conceptually, a set of recommendations can also be viewed as *choice query*, in which the user indicates which option is most preferred from that set.

Because of the potential tension between making a good set recommendation and asking an informative choice query, we explore the connection between the two. We consider two different models of preference uncertainty and optimization: (a) a Bayesian framework in which a posterior over user utility functions is maintained, optimal recommendations are assessed using expected utility, and queries are assessed using expected value of information; and (b) a minimax-regret framework in which user utility uncertainty is strict (represented by a polytope), recommendations are made using the minimax-regret robustness criterion, and queries are assessed using worst-case regret reduction. We show that, somewhat surprisingly, in both cases, there is no tradeoff to be made between good recommendations and good queries: we prove that the optimal recommendation set of size k is also an optimal choice query of size k. We also examine the case where user responses to choice queries are error prone (using both constant and mixed multinomial logit noise models) showing the results are robust to this form of noise.

In both frameworks, our theoretical results have practical consequences for the design of interactive recommenders. Our results also allow us to design efficient algorithms to compute optimal query/recommendation sets. We develop several such algorithms (both exact and approximate) for both settings and provide empirical validation of their performance.

Keywords: preference elicitation, utility elicitation, preferences, minimax regret, Bayesian inference, utility theory, decision-making, recommender systems, value of information, submodularity.

[☆] Parts of this article appeared in: Regret-based Optimal Recommendation Sets in Conversational Recommender Systems, *Proc. of the ACM Conference on Recommender Systems (RecSys-09)*, pp.101–108, New York, 2009 [131]; Optimal Bayesian Recommendation Sets and Myopically Optimal Choice Query Sets, *Advances in Neural Information Processing Systems 23 (NIPS-10)*, pp.2352–2360, Vancouver, 2010 [132]; and Recommendation Sets and Choice Queries: There Is No Exploration/Exploitation Tradeoff!, *Proc. of the Twenty-fifth AAAI Conference on Artificial Intelligence (AAAI-11)*, NECTAR track, pp.1571–1574, San Francisco, 2011 [133].

[*] Corresponding author

Email addresses: `paolo.viappiani@lip6.fr` (Paolo Viappiani), `cboutilier@google.com` (Craig Boutilier)

URL: `http://www-desir.lip6.fr/~viappianip/` (Paolo Viappiani), `http://www.cs.toronto.edu/~cebly/` (Craig Boutilier)

[1] Part of this work was completed while the author was at the Department of Computer Science, University of Toronto, Canada.

[2] Work conducted while at the Department of Computer Science, University of Toronto, Toronto, Canada.

1. Introduction

Artificial intelligence (AI) technologies have become ubiquitous in the recommendation of products, services, information and enterainment content, and in a variety of decision support systems. As users come to rely increasingly on automated recommendations, the development of methods to effectively and accurately assess user preferences is needed to ensure that AI-based recommenders adequately reflect the interests of the users on whose behalf they act. This need is especially pressing as decision support systems and recommenders begin to take autonomous actions on behalf of users rather than simply "recommending" alternatives to them.

The assessment of user preferences can be either *passive* or *active*. In passive systems, recommendations are made directly to meet the recommender's *estimated* preferences of the user. The user's response offers clues to her underlying preferences or utility function, and is typically used to improve that estimate. In many content recommenders (e.g., as used for movies, music, video, news, etc.), the repeated consumption of multiple items by users allows *collaborative filtering (CF)* techniques to be applied [64, 106, 110, 47, 77]. However, their passive nature means that CF methods rarely take active steps to discern a user's preferences—rather they emerge as simply a by-product of interaction with the recommendations. In other domains, such as multi-attribute product recommendation, where a single item is to be (say) purchased, passive recommenders will often recommend one or several products, and the user will provide some feedback (e.g., propose a different attribute value or accept a specific recomendation). For example, *conversational* and *critiquing* recommenders often work in this fashion [29, 91, 35, 122].

By contrast, *preference elicitation* techniques are more active, explicitly asking users queries about their preferences or utility functions [18, 26, 32, 1, 12, 123, 13]. While the use of explicit preference elicitation has primarily focused on single-recommendation domains such as multi-attribute product search [69, 1, 25, 27, 59, 17, 13], it has also received attention in collaborative filtering style recommenders [23, 48, 83]. It also plays a role in other AI applications (e.g., planning systems [6, 101, 139, 52, 99], search [11], social choice [86, 94, 10] and interface agents [50]). Active elicitation methods typically continue to ask questions until enough user preference information is extracted to provide some guarantees on the quality of an "ultimate" recommendation.

Naturally, approaches to recommendation and elicitation can incorporate elements of both passive and active elicitation. For example, many conversational or critiquing systems recommend a small set of items, but often select those items not only based on their estimated preference, but also based on characteristics about which the user may provide direct feedback [134, 143, 34, 45]. In this work, we consider recommenders from this dual active-passive perspective, in which the set of items presented to the user can serve as both recommendations—or *choice sets*—and as *choice queries* which allow users to express their preference for one item over the others without serving as a final recommendation.

When viewed from this dual perspective, a fundamental tension seems to exist. On the one hand, a recommender should propose a an optimal *recommendation set*, that is, a set of items from which one is likely to have high expected utility for the user (with respect to the recommender's estimate of user preferences). In other words, we desire recommendation sets that are *exploitive*. On the other hand, the recommender should propose an optimal *choice query* with high expected value of information; that is, a set of items such that when the user states which she prefers, the recommender's *updated* estimate of the user's preference (w.r.t. that response) maximally improves the recommender's ability to make a good recommendation. In other words, we desire choice queries that are strongly *exploratory*. These two demands on the presented set of items seem to be at odds.

In this work, we examime this prospective tension directly, and demonstrate, somewhat surprisingly that there is in fact *no exploration-exploitation tradeoff* in set recommenders under a variety of conditions, as we elaborate.

1.1. Elicitation and Uncertainty Modeling

We first provide a high-level view of two different styles of uncertainty modeling, commonly used on adaptive preference elicitation [32, 18, 26, 59, 13], under both of which we analyze the exploration-exploitation tradeoff. Abstractly, most techniques for generating queries use some form of *belief state* to capture the recommender's knowledge, and degree of uncertainty. This belief state is updated given user responses to queries or observed choice behavior. If we treat making a recommendation as a "query," the typical elicitation scheme proceeds as follows:

1. Initialize the belief state given some initial user preferences or prior.

2. **Repeat** until the belief state meets some termination condition:

   (a) Ask user a query q.
   (b) Observe the user response r.
   (c) Update the belief state given r.

3. Recommend the item optimal according to the current belief state.

The termination condition might be internal to the recommender (e.g., when some notion of "recommendation uncertainty" drops below some threshold) or external (the user is satisfied by, selects, or otherwise "commits" to a proposed item). The efficiency of the general elicitation method described above depends crucially on the system's ability to select good queries. Given the current belief about the user's utility function, "good" queries are such that the quality of future recommendations can quickly improve.

There are two main frameworks for representing utility or preference uncertainty (i.e., the belief state).

Regret-based elicitation. In *regret-based elicitation*, the recommender maintains an explicit representation of a set of feasible utility functions, usually represented compactly by constraints on parameters; recommendations are generated using the *minimax regret criterion* [113, 8, 84, 21]. Minimax regret provides a robust way of making decisions under utility function uncertainty, minimizing worst-case loss under all possible realizations of a user's utility function; as such it is applicable when distributional information over utility functions is not easily available. It can be thought of as the solution of a game between the recommender system and a hypothetical adversary, with the former choosing an item and the latter choosing a feasible utility function in order to maximize *regret*, that is, the utility difference between the true best item and the recommender's choice.

Regret can also act as an efficient driver for elicitation: the result of the minimax regret computation can suggest which query to ask next. For example, the *current-solution strategy (CSS)* for selecting queries (asking to compare the regret-optimal item with the alternative maximizing the utility function picked by the adversary) is effective in practice; however CSS provides no strong theoretical guarantees about the reduction of minimax regret *a posteriori* (after incorporating the user response in the belief, and producing a new recommendation in the next interaction cycle). A method that may produce more informative queries is to directly adopt the *worst-case a posteriori regret* as a criterion for query selection: queries are evaluated according to the smallest reduction of regret with respect to all possible user responses; this approach is in general more computationally expensive than CSS.

Bayesian elicitation. In *Bayesian elicitation* [32, 18], a probabilistic prior distribution over user utility parameters is assumed. The distribution is updated using Bayes' rule whenever new information is acquired from the user (e.g., query responses or behavioral observations). Whenever a single recommendation has to be made, the option with highest expected utility is recommended. The most natural criterion for choosing queries is *expected value of information (EVOI)*, which can be optimized myopically [32] or sequentially [18]. However, optimization of EVOI for online query selection is not feasible except in simple cases. Hence, in practice, heuristics are used that offer no theoretical guarantees with respect to the quality of the query.

Recommendation Sets and Choice Queries. In this work, we focus on *choice queries*; such queries are commonly used in conjoint analysis and product design [85] and require a user to indicate which choice/product is most preferred from a set of k options. We also focus on *recommendation sets*—rather than recommending a single item, we suppose the recommender can suggest a set of k items from which the user can select her most preferred.

1.2. Our Contributions

In this work, we address the question of generating the most informative query given the current belief about the user's utility function, within both the regret-based and Bayesian frameworks. We adopt *expected value of information (EVOI)* as the criterion for evaluating queries in the Bayesian framework and *worst-case regret* in the regret-based framework. Such criteria can often generate better queries than strategies that aim at reducing uncertainty *per se*, such as entropy-based methods, or heuristics based on product "diversity" [69, 1, 126, 103, 143, 2].

In both settings we show how (myopically) optimal or near-optimal queries can be generated by exploiting the connection with the problem of generating an optimal recommendation set. In particular, we show that, under very general assumptions, optimization of choice queries reduces to the (prima facie) simpler problem of choosing the *optimal recommendation set*, i.e., the set of k items such that, if a user were forced to choose one, maximizes utility of that choice (under utility uncertainty).

This has a number of implications:

- The most significant consequence of this fact is that any set-based recommender that adopts the dual perspective of generating sets that are maximally informative (i.e., from an exploratory point of view) and generating sets that have maximal immediate expected utility for the user (i.e., from an exploitive point of view) need not make decisions about which viewpoint to adopt at different points in time. The same set of items accomplishes both. In other words, there is no exploration-exploitation tradeoff.
- In both frameworks, optimizing a recommendation set is a computationally simpler problem than direct optimization of optimal choice queries using "straightforward" algorithms. As a consequence, this equivalence allows us to demonstrate that generating choice queries is computationally easier than explicitly computing EVOI.
- In the Bayesian framework the optimal recommendation set problem is submodular, admitting a greedy algorithm with strong approximation guarantees. Thus, it can be used to determine approximately optimal choice queries. We develop this connection under several different (noisy) user response models. We describe *query iteration*, a local search technique that, though it has no formal guarantees, finds near-optimal recommendation sets and queries much faster in practice than either exact or greedy optimization.
- In the regret-based framework, our methods can be used to efficiently optimize queries and produce queries that guarantee a higher (myopic) reduction of regret (in the worst-case) relative to the current-solution strategy. The regret-based variant of query iteration can be used to improve query sets incrementally starting from a seed set. When the current solution is used as a seed, the resulting set is provably no worse than the current-solution strategy.

Our work has significant practical implications. In many settings, we may wish to present a set of options to a user with the dual goals of offering a good set of recommendations and eliciting valuable information about the user's utility over item space. For instance, product navigation interfaces for e-commerce sites often display a set of options from which a user can select, but also give the user a chance to critique the proposed options [33, 134]. This provides one motivation for exploring the connection between optimal recommendation sets and optimal choice query sets. Moreover, we show that our findings are useful even in settings where queries and recommendation are clearly separated, as the computation of the most informative query to ask can be made more efficient by exploiting the theoretical connection between the two concepts.

1.3. Outline

The paper is organized as follows. We first present the background in Section 2, we further motivate our work, discussing the role of item sets in recommendation systems, and introducing the two problems that we tackle in this paper: the generation of recommendation sets and the generation of choice queries in order to acquire relevant information (Section 2.1). We provide basic technical background on multiattribute utility theory (Section 2.2), the representation of utility function uncertainty (Section 2.3), and on the two main elicitation frameworks of interest: regret-based (Section 2.4) and Bayesian (Section 2.5).

We develop our theoretical contributions by considering each framework in turn, starting with the regret-based framework in Section 3. We first consider recommendation sets and introduce a new decision criterion, *setwise max regret*, that generalizes minimax regret to cover the case of recommendation sets under strict utility uncertainty (Section 3.1) and formalize a notion of myopic value of information to assess the quality of a choice query (Section 3.2). Our main technical result is presented in Section 3.3, showing that the optimal recommendation set is also an optimal choice query. We provide a detailed example in Section 3.4 and conclude with algorithms (Section 3.5) for generating sets of items to be used for set recommendation and/or elicitation with choice queries.

In Section 4, we adopt the Bayesian framework and proceed in a similar way. We start by formalizing the notions of recommendation sets from a Bayesian point-of-view (Section 4.1) and elicitation with choice queries (Section 4.2). In order to perform Bayesian inference over user utility functions, we need a probabilistic model of user responses, so we analyze some commonly used response formats: noiseless responses, a constant noise model, and a logistic noise model (response models and their impact on recommendation are discussed in Sections 4.3 and 4.4). In Section 4.5, we develop the theory for the Bayesian framework in two stages. We first deal with noiseless responses, showing the equivalence of optimal recommendation sets and myopically optimal query sets, then provide similar results for the noisy response models. We provide detailed examples in Sections 4.6 and develop efficient algorithms for computing recommendation/query sets, including approximation algorithms with theoretical guarantees in Sections 4.7.

Section 5 provides empirical validation with simulations of both frameworks. After presenting the general methodology (Section 5.1), we assess the performance of elicitation in regret-based (Section 5.2) and Bayesian frameworks (Section 5.3). Finally, we propose some directions for future research, along with connections to additional related work in Section 6, before concluding in Section 7.

2. Background

In this section we elaborate on the motivation for set-based recommendation and querying in recommender systems and outline both the regret-based and Bayesian formal frameworks within which we cast our subsequent analysis and results. We also introduce several new concepts that figure prominently in our analysis.

2.1. Utility-theoretic Preference Elicitation and Item Sets

Preferences are a crucial bottleneck in many AI applications: in order to provide personalized services on the behalf of a user, an automated system must have a good model of the user's preferences. However, systems must often make decisions with limited preference information since a complete assessment of user preferences is usually impractical. This is due to the fact that acquiring preference information (e.g., by asking the user to rate additional items, compare items, or ask questions about item attributes) is often expensive, both with respect to time and cognitive burden.

In this work, we focus on principled recommendation and elicitation approaches based on user *utility*. We assume that a user has an underlying "true" *utility function* over items in the domain that dictates how she makes choices, whether answering preference queries or accepting or rejecting recommendations. Since the recommender will not know this underlying utility function, we consider the problem of *interactive utility elicitation*, where the system must decide which queries are most informative relative to the goal of making good or optimal recommendations.

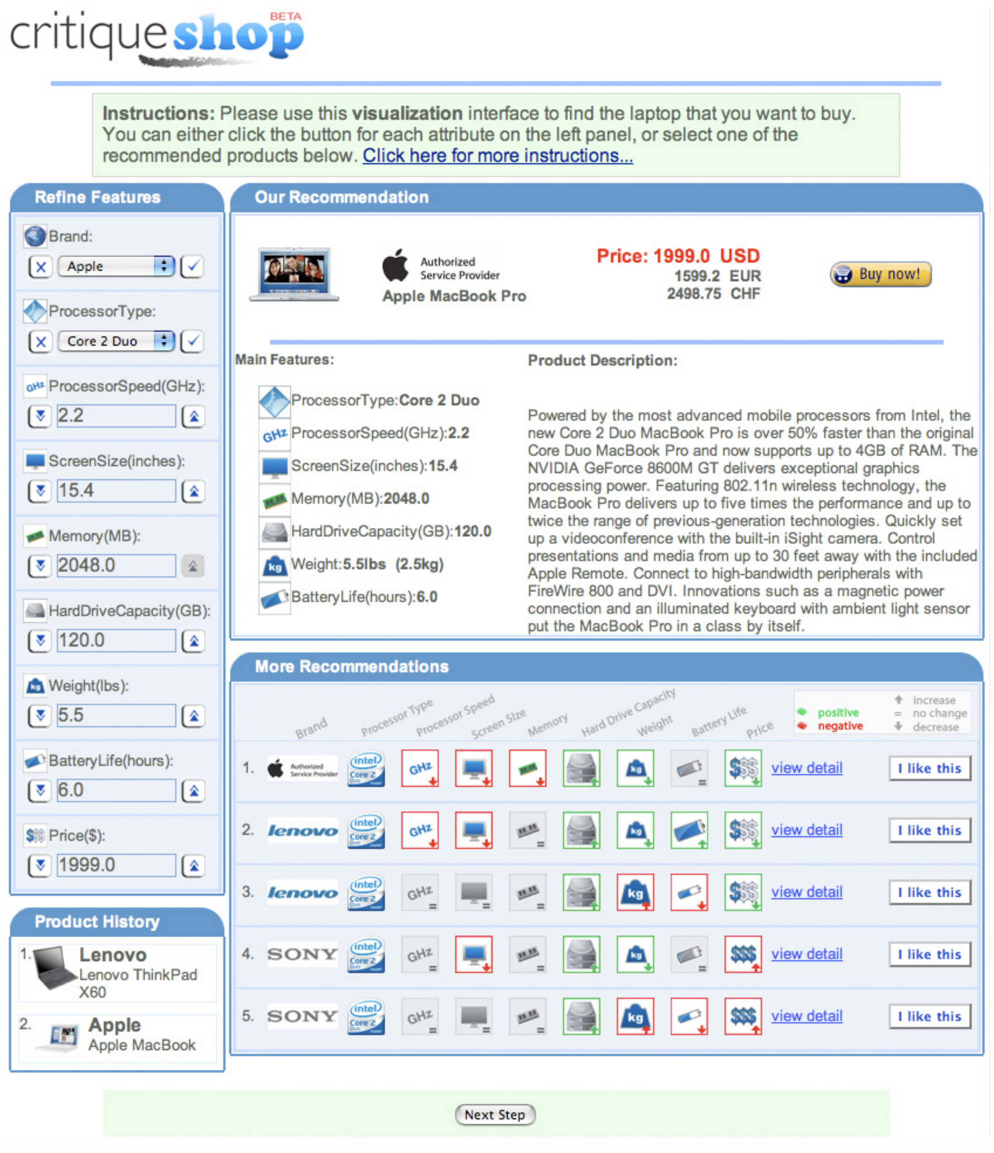
The utility-theoretic perspective should be viewed as a methodological convenience. It is widely acknowledged that users have a hard time assessing their own utilities, especially in quantitatively precise terms [128]. In this work, we address this by focusing on queries and choices that involve qualitative comparisons and allow for noise in a user's responses. Considerable work in behavioral economics [30] also suggests that users often *construct* (some of) their preferences as they engage in decision making rather than revealing latent utilities; indeed, facilitating such preference discovery and construction is an important role for recommenders in AI. While refinements of elicitation methods that incorporate such considerations are important, this does not obviate the need for techniques like ours that uncover these preferences (whether constructed or revealed). The utility-theoretic perspective also stands in contrast with qualitative models based on partial orders such as CP-nets [20], where options or choices can be inherently incomparable.

A variety of principled approaches have been proposed for utility elicitation. A number of these focus directly on (myopically or heuristically) reducing uncertainty regarding utility parameters as quickly as possible, including max-margin [50, 123], volumetric [69], polyhedral [126] and entropy-based [1] methods. However, we do not attempt to reduce utility uncertainty for its own sake, but rather focus on discovering utility information that improves the quality of the recommendation—indeed, this is a key ingredient in our theoretical approach.

In this paper we address the use of sets of items both as elicitation queries and as recommendation sets:

- Generating a set of recommendations for the user given the known preference information, and
- Selecting a choice query in order to acquire relevant preference information.

As discussed above, this induces a potential tension in the form of an exploration-exploitation tradeoff, that is, between making good recommendations for the user and eliciting "useful" information from the user. One might expect the


Instructions: Please use this **visualization** interface to find the laptop that you want to buy. You can either click the button for each attribute on the left panel, or select one of the recommended products below. Click here for more instructions...

Refine Features

Brand:	Apple
ProcessorType:	Core 2 Duo
ProcessorSpeed(GHz):	2.2
ScreenSize(inches):	15.4
Memory(MB):	2048.0
HardDriveCapacity(GB):	120.0
Weight(lbs):	5.5
BatteryLife(hours):	6.0
Price($):	1999.0

Product History

1. Lenovo — Lenovo ThinkPad X60
2. Apple — Apple MacBook

Our Recommendation

Authorized Service Provider
Apple MacBook Pro

Price: 1999.0 USD
1599.2 EUR
2498.75 CHF

Buy now!

Main Features:

- ProcessorType:**Core 2 Duo**
- ProcessorSpeed(GHz):**2.2**
- ScreenSize(inches):**15.4**
- Memory(MB):**2048.0**
- HardDriveCapacity(GB):**120.0**
- Weight:**5.5lbs (2.5kg)**
- BatteryLife(hours):**6.0**

Product Description:

Powered by the most advanced mobile processors from Intel, the new Core 2 Duo MacBook Pro is over 50% faster than the original Core Duo MacBook Pro and now supports up to 4GB of RAM. The NVIDIA GeForce 8600M GT delivers exceptional graphics processing power. Featuring 802.11n wireless technology, the MacBook Pro delivers up to five times the performance and up to twice the range of previous-generation technologies. Quickly set up a videoconference with the built-in iSight camera. Control presentations and media from up to 30 feet away with the included Apple Remote. Connect to high-bandwidth peripherals with FireWire 800 and DVI. Innovations such as a magnetic power connection and an illuminated keyboard with ambient light sensor put the MacBook Pro in a class by itself.

More Recommendations

1. Apple — view detail — I like this
2. lenovo — view detail — I like this
3. lenovo — view detail — I like this
4. SONY — view detail — I like this
5. SONY — view detail — I like this

Next Step

Figure 1: CritiqueShop, designed and developed by Reilly et al. [104], is an electronic commerce tool where suggested options are shown with the dual goal of elicitation and recommendation. The left panel shows the initial preferences. The current recommendation is shown at the top, while a set of alternative suggested products is shown below (each corresponding to a tradeoff between features that are improved with respect to the current recommendation, marked by a green rectangle, and features that are downgraded, marked by a red rectangle); whenever the user click on "I like this" next to a product, this becomes the new current recommendation in the new cycle. The internal preference model is also updated to incorporate the acquired tradeoff that has an impact in the suggested set displayed next.

solutions to these two problems to be radically different. Intuitively, in order to make a good recommendation the system should exploit its current knowledge of the utility function. On the other hand, when asking queries, the system aims to acquire more information, so better recommendations can be made in the future.

Since utility is uncertain, there is often value in recommending a *set* of options from which the user can choose her most preferred. Picking a diverse set of recommended options increases the odds of recommending at least one item with high utility. Intuitively, such a set of "shortlisted" recommendations should include options that are *diverse* in the following sense: recommended options should be highly preferred relative to a wide range of "likely" user utility functions (relative to the system's current belief) [23, 100].

In broad terms, the system must identify a set of items that maximizes some form of expected valuation of the user's choice from the recommended set. Our set-based optimization approach passes some residual uncertainty (about the preferences for items in the set) back to the user to resolve [100]. This stands in contrast to some recommender systems that define diversity relative to product attributes [102, 103, 145, 2, 130, 79] with no direct reference to beliefs about user utility. It is not hard to see that "top k" systems, those that present the k options with highest expected utility, do not generally result in good recommendation sets [100].

Resolving this exploration-exploitation tension is also important because many recommenders present options with dual goal of recommendation and elicitation. This is common, for instance, in conversational recommender systems such as *CritiqueShop* [104] (see Figure 1), where several products are displayed, and user selections do not only update the preference model, but might also lead to purchase decisions. Since a user can end the interaction unpredictably at any time, it is important that the system show products that are likely to be highly preferred.

Among the many possible types of queries, we focus on *choice queries*. Such queries are commonly used in conjoint analysis and product design [85], requiring a user to indicate which choice/product is most preferred from a set of k options. Since we focus on choice queries, we can view any set of products as either a recommendation set or query (or choice) set. Given a set, one can ask: *what is the value of the set viewed as recommendation set ?*; and *what is its value as a query?* Choice queries are generalizations of comparison queries, where $k = 2$ and a user is asked if they prefer one item over another.

In this work we develop the connection between the value of a set as a recommendation and as a choice query, in both the Bayesian and the regret-base frameworks. In both cases we show that, somewhat surprisingly, and under very general assumptions, the optimal recommendation set is also an optimal choice query. These results are intriguing theoretically and have important practical implications.

We formalize the problem of generating a recommendation set and selecting a choice query in both the regret-based and Bayesian frameworks. There are some analogies between the theoretical models and some common elements in the construction of the proofs of our main results. We provide decision and elicitation criteria for both cases that extend classical decision criteria to sets, as summarized in the following table:

	Framework	
Value of...	Bayesian	Regret-based
a single recommendation	Expected Utility	Minimax Regret
a recommendation set	Expected Utility of Selection (EUS)	Minimax *Setwise* Regret
a query	Expected posterior Utility (EPU)	Worstcase Regret (WR)

Naturally, each treatment of utility uncertainty has its own advantages and drawbacks. The advantage of a Bayesian approach is its ability to easily exploit prior information, reason about the likelihood of user responses, support "noisy" responses, and produce recommendations that are optimal in expectation. These advantages do come at a price—inference can be computationally expensive, while acquisition of priors is often costly and may not even be available in many circumstances. By contrast, regret-based elicitation can provide robust recommendations even when (quantitative) priors are not available. Moreover, the regret-based model can provide guarantees on the utility of a recommendation that are are sometimes stronger than those offered by expected utility. We do not argue for one approach here, but rather construct a methodology for utility elicitation using optimal recommendation sets, and develop corresponding algorithmic methods for both frameworks. Finally, we note that do not study the relationship between the Bayesian and regret-based approaches in any depth; but we develop both within the *same* article because of the strong parallels in the structure of our analyses and algorithms in each case.

2.2. Multiattribute Utility Models

We assume a system is charged with the task of recommending an option to a user in some multiattribute space, for instance, the space of possible product configurations from some domain (e.g., computers, cars, rental apartments, etc.). Products are characterized by a finite set of attributes $\mathcal{X} = \{X_1, \ldots, X_n\}$, each with finite domain $Dom(X_i)$. Let $X \subseteq Dom(\mathcal{X})$ denote the set of *feasible configurations*. For instance, attributes may correspond to the features of various cars, such as color, engine size, fuel economy, etc., with X defined either by constraints on attribute combinations (e.g., constraints on computer components that can be put together) or by an explicit database of feasible configurations (e.g., a rental database).

The user has a *utility function* $u : Dom(\mathcal{X}) \to \mathbb{R}$. The precise form of u is not critical to our theoretical results. We assume that u is parameterized by some parameter (or weight) vector $w \in W$, where W is the space of feasible parameter vectors. We write $u(x; w)$ to emphasize this dependence. In such a case, we often refer to w as the user's "utility function" for simplicity, assuming some fixed form for u across the user population. As we will see later, efficient implementations of our methods are possible when $u(x; w)$ is linear in the parameters (or weights) w (e.g., as in generalized additive independent (GAI) models [49, 25]). A simple *additive model* in the car domain (where x is a car with features $MPG, EngineSize, Color$) might be:

$$u(x; w) = w_1 x[MPG] + w_2 x[EngineSize] + w_3 x[Color].$$

An *optimal item* x^*_w for a user with utility w is any item in X that maximizes $u(x; w)$.

Alternative, more complex, utility models depart from the assumptions of linear utility. Some models are non-linear with respect to item attributes, but still are effectively linearly parameterizable. Others are inherently non-linear. What is important is that the structure (i.e., the parametrization) of the utility function is given. Our framework does not depend on the type of structure, only that it is given (learning or assessing the structure is not in scope in this paper). For example, the following non-linear parametrizations can be incorporated in our framework (assume items are defined on m features; represented as $x[1], ..., x[m]$):

- Multiplicative utility: $u(x; w) = \prod_i x[i]^{w_i}$.
- Sum-log: $u(x; w) = \sum_i w_i log(x[i])$.
- Ideal point model: $u(x; w) = \sum_i (x[i] - w_i)^2$.
- Ordered Weight Average (OWA) [142]: $u(x; w) = \sum_i w_i x[(i)]$ where (i) is the permutation such that $x[(i)] \leq x[(i+1)]$ for all $i = 1, ..., m$ with $x[(i)]$ being the ith lowest feature.

We refer the reader to Torra and Narukawa [125] and Grabisch et al. [57] for a detailed analysis of different utility (or aggregation) functions.

2.3. Utility Function Uncertainty and Elicitation

In general, a recommender will possess incomplete information about a user's utility function. Since we assume that the utility function is uniquely identified by the parameter vector w, information about the utility function can be directly equated with information about w. For this reason, we abuse terminology and sometimes refer to w itself as the utiity function. This information, hence the recommender's uncertainty surrounding w, will be represented (and possibly compressed, abstracted, summarized or approximated) in some fashion. We refer to this representation as the recommender's *belief state* about the user's preferences and denote it abstractly by θ. We discuss specific forms of θ below.

If a recommender is required to make a recommendation when it is uncertain about w, it needs some specific criterion for making decisions in the face of such uncertainty. Without direct access to w, it cannot generally recommend x^*_w; instead it should recommend $x^*_\theta \in \arg\max_{x \in \mathcal{X}} s(x; \theta)$, the best item given its belief θ about w under some suitable evaluation criterion or "score" s. This criterion will depend on the representation itself—we discuss several criteria below. Naturally, the scoring function should have a tight connection to the underlying true utility $u(x^*_\theta; w)$ the user derives from the recommended item.

As discussed above, when there is uncertainty about w, recommending a set of k items $S = \{x_1, \dots x_k\}$ to a user so that she can select her most preferred from that set can improve overall recommendation quality relative to recommending a single item. For example, if we assume that the user will select her most preferred item from the set, we can cover different "modes" of uncertainty in θ as we elaborate below. We refer to S as a *recommendation set*.

The recommender can refine its belief state θ by learning more about w. A reduction in uncertainty should lead to better recommendations (in expectation). While many sources of information can be used to assess the preferences of a user —including the preferences of related users, as in collaborative filtering [76, 47], or observed user choice behavior [85, 97]—we focus on explicit *utility elicitation*, in which a user is asked questions about her preferences.

There are a variety of query types that can be used to refine one's knowledge of a user's utility function (we refer to [74, 21, 25] for further discussion). *Comparison queries* are especially natural, asking a user if she prefers one option x to another y. These comparisons can be localized to specific (subsets of) attributes in additive or GAI models, and such structured models allow responses w.r.t. specific options to "generalize," providing constraints on the utility of related options. In this work we consider the extension of comparisons to *query sets* of more than two options [131, 132] as is common in conjoint analysis [85, 126]. Any set S of $k > 1$ items can be interpreted as a query: the user states which of the k elements $x_i \in S$ she prefers. We refer to S as a *choice query* in this context. We will use choice query and recommendation set interchangeably to describe any set S as warranted by context; and some cases below, a set S will play both roles simultaneously.

Given a recommendation set S with $x \in S$, we adopt the following notation to represent preferences and user choice among items in S:

- $S \rhd x$ denotes that x has maximum utility among those items in S, given some fixed utility function w, i.e., $x \in \arg\max_{x \in S} u(x; w)$
- $S \rightsquigarrow x_i$ denotes the event of the user selecting x_i among the items in S.

It is useful to explicitly partition the utility space W based on which item in S has highest utility.

Definition 1 *Given utility space W and set $S = \{x_1, \dots, x_k\}$, the* S-cover *of W is the collection $\{W_{S \rhd x_i} | \forall i \in \{1, \dots, k\}, W_{S \rhd x_i} \neq \emptyset\}$, where $W_{S \rhd x_i} = \{w \in W | u(x_i; w) \geq u(x_j; w), \forall x_j \in S\}$.*

Set $W_{S \rhd x}$ contains all $w \in W$ such that x has utility under w that is at least that of any other option in S.

Since a set $S = \{x_1, \dots, x_k\}$ may contain dominated alternatives, the S-cover may have cardinality less than k. For instance, if $S = \{x_1, x_2, x_3\}$ and the utility of x_2 is less than that of either x_1 or x_3 for all $w \in W$, then the S-cover is $\{W_{S \rhd x_1}, W_{S \rhd x_3}\}$. Note that an S-cover always exists; and if a single item x_1 dominates all others in S, then the partition is just $\{W_{S \rhd x_1}\} = \{W\}$.

When a user answers a choice query S by stating which $x_i \in S$ is her most preferred option, the recommender will update its belief state θ. If we assume that she answers accurately with respect to her underlying true utility w, then the choice of $x_i \in S$ refines the set of feasible utility functions W by imposing $k - 1$ constraints of the form $u(x_i; w) \geq u(x_j; w), j \neq i$ that should be incorporated into θ. When we discuss Bayesian elicitation, we also allow the possibility of "noisy" responses in which this answer may not guarantee that such constraints are valid.

2.4. Regret-based Recommendation and Adaptive Elicitation

We first elaborate on a model for recommendation and preference elicitation when beliefs about utility reflect *strict uncertainty*; in other words, the utility uncertainty is represented by constraints on the parameters of the utility function, with no further quantification of how likely specific utility functions are. We adopt *minimax regret* [114] as our decision criterion for robust decision making under such utility function uncertainty. Minimax regret has been introduced as a means for robust optimization in the presence of data uncertainty [78], and has more recently been advocated for decision making with utility uncertainty [111, 21].

The *maximum (or max) regret* of option (or choice) x is the maximum loss (in term of utility) that can be incurred by not choosing the (unknown) true optimal choice. This regret-based approach has several advantages. First, the recommender's belief state θ is easy to update: whenever a query is answered, it imposes a new preference constraint from which one derives a refined feasible set of utility functions (usually imposing a constraint on the parameters w). Second, simple "prior" information (e.g., preference information that is available before elicitation begins) can be encoded with constraints in the space of utility parameters (for instance, in the car domain, requiring that the weight for "MPG" is higher than the weight for "luggage capacity"). Third, there are efficient heuristics that directly use the computation of minimax regret to choose queries to ask the user.

Minimax regret has proven to be an effective tool for recommendation and interactive elicitation for a variety of utility models. In addition to linear utility models, regret-based elicitation has been applied to the elicitation of GAI models [25] and utility models based on the Choquet integral [12]. Its main limitation is the inability to deal naturally with noisy responses.

2.4.1. Robust Optimization for Generating Recommendations

Assume that through some interaction with a user, and possibly using some prior knowledge, we determine that her utility function w lies in some set W. This will serve as the recommender's belief state (and will evolve as it engages in elicitation). The form of W will become clearer when we discuss elicitation below. We assume that W is topologically closed and compact.

Following Boutilier et al. [21], we define minimax regret in three steps. We first define the regret of an item under a known utility function. We then define pairwise max regret (the worst-case loss associated with choosing an item instead of another one), the max regret of an item (the worst-case loss associated to not choosing the true best item), and finally minimax regret and the associated regret-optimal item.

Definition 2 *Given utility function $w \in W$, the* regret $R(x; w)$ *of item (or recommendation) x is the loss in utility associated with choosing x over the optimal item:*

$$R(x; w) = \max_{y \in X} u(y; w) - u(x; w) = u(x^*_w; w) - u(x; w).$$

Definition 3 *Given a set of utility functions W and $x, y \in X$, the* pairwise max regret $\mathrm{PMR}(x, y; W)$ *of x w.r.t. y is*

$$\mathrm{PMR}(x, y; W) = \max_{w \in W} u(y; w) - u(x; w). \tag{1}$$

Definition 4 *The* max regret $\mathrm{MR}(x; W)$ *of $x \in X$ over feasible utility functions W is*

$$\mathrm{MR}(x; W) = \max_{y \in X} \mathrm{PMR}(x, y; W) = \max_{y \in X} \max_{w \in W} u(y; w) - u(x; w) = \max_{w \in W} R(x; w). \tag{2}$$

$\mathrm{MR}(x; W)$ is the worst-case loss associated with recommending item x. Intuitively, one can view this as an adversary choosing w from W to maximize the difference in utility between the optimal item (under w) and x.

Definition 5 *The* minimax regret $\mathrm{MMR}(W)$ *of W and the* minimax regret optimal item x^*_W *are defined as*

$$\mathrm{MMR}(W) = \min_{x \in X} \mathrm{MR}(x; W) \tag{3}$$

$$x^*_W \in \arg\min_{x \in X} \mathrm{MR}(x; W). \tag{4}$$

Minimax regret requires that the recommender propose an item that minimizes worst-case loss (i.e., max regret) w.r.t. W. The minimax optimal item x^*_W minimizes this potential loss, while $\mathrm{MR}(x; W)$ bounds the loss associated with x, and is zero if and only if x is optimal for all $w \in W$. Any choice that is not minimax optimal has strictly greater loss than x^*_W for some $w \in W$.

Minimax regret relies on relatively simple prior information in the form of bounds or constraints on user preferences (rather than probabilistic priors); and exact computation can be perfomed efficiently using various optimization techniques. In configuration problems, optimization over product space X is often formulated as a constraint satisfaction problem (CSP) or mixed integer program (MIP). In such domains, minimax regret computation can be formulated as a MIP, and solved practically for large problems using techniques such as Bender's decomposition and constraint generation [21, 22, 25]. When instead the available choices are stored in a database, minimax regret can be computed implementing a search using alpha-beta cuts [24].

2.4.2. Regret-based Utility Elicitation

The minimax regret criterion can be used in an incremental elicitation process that progressively asks queries to the user in order to assess preference parameters with increasing accuracy until MMR drops below a given threshold (alternatively, other termination criteria can be adopted, for example, when the elicitation cost becomes too high, or termination might be left in the hands of the user).

Indeed, note that minimax regret MMR cannot increase when adding constraints to W (in particular, when incorporating new preference information about the user) and it usually decreases (see [24], pages 194-202). The proof is straightforward and relies on the fact that when the adversary has less choice in picking the adversarial instantiation of the weights under which regret is computed, the regret must necessarily be smaller or the same. We list some simple observations.

Observation 1 *Given two sets of utility functions W, W' such that $W' \subseteq W$, the following inequalities hold:*

- $\mathrm{PMR}(x, y; W') \leq \mathrm{PMR}(x, y; W)$, $\forall x, y \in X$
- $\mathrm{MR}(x; W') \leq \mathrm{MR}(x; W)$, $\forall x \in X$
- $\mathrm{MMR}(x; W') \leq \mathrm{MMR}(x; W)$

Queries can be of many different types. For instance, *bound queries* ask if the overall utility (aggregate value) of an alternative is higher or lower than a given reference value (or some "willingness to pay"). Comparison queries are relatively cognitively easy to answer as they require the user to compare a pair of alternatives and state which one is preferred. Even assuming that the type of the query is fixed, there are of course several queries that one can ask at each step.

Note that some queries will be more informative than others. For instance, asking to compare one alternative with one another that is Pareto-dominated by the first will provide no value (MMR will not change); instead asking to compare two potentially good alternatives will possibly constitute a good query. Choosing good queries is therefore essential to providing good recommendations reasonably quickly.

Assume that, at a given point in the interaction, the user's preferences induce a space W of possible utility parameters. Given a query q and its possible responses $\mathcal{R}_q$, let W_r denote the parameter space that results when the user answers r, for any $r \in \mathcal{R}_q$. Note that $W_r \subseteq W$. A good query will significantly reduce MMR for each possible response (recall that, by Obs. 1, for any $r \in \mathcal{R}_q$, $\mathrm{MMR}(W_r)$ can no greater than $\mathrm{MMR}(W)$). A non-probabilistic notion of *myopic value of information* can be defined as follows (generalizing the pairwise measure of [21]):

Definition 6 *The myopic* worst-case a posteriori regret (WR) *of a query q is*

$$\mathrm{WR}(q; W) = \max_{r \in \mathcal{R}_q} \mathrm{MMR}(W_r)$$

where W_r is the subset of W obtained by adding the constraints encoding r. Given some set of queries Q, a WR-optimal query *is*

$$q^*_W \in \arg\min_{q \in Q} \mathrm{WR}(q; W). \tag{5}$$

Intuitively, WR can be interpreted as the value of information carried in a query q assuming that a user provides a feasible response that leaves as much regret as possible in the resulting belief state W_r. This is equivalent to considering the worst-case (over possible responses) *regret reduction*: $\min_{r \in \mathcal{R}_q} \left\{ \mathrm{MMR}(W) - \mathrm{MMR}(W_r) \right\}$. The WR-optimal query q^*_W offers the maximum worst-case regret reduction.

Given the computational demands of maximizing a score such as WR, it is quite natural to consider heuristics for query generation. The *current solution strategy (CSS)*, first described in [22, 21] generates a query that involves utility parameters in either x^*_W (the minimax-regret optimal item) or x^a (the adversarial item), or both. This is based on the insight that should a query provide information about no parameter that impacts the utility of either x^*_W or x^a, minimax regret is unlikely to change. When considering comparison queries, CSS proposes a comparison between x^*_W and x^a. Note that asking queries according to the CSS cannot lead to inconsistencies, since the adversarial item x^a at any stage must be optimal for some $w \in W$. As a consequence, elicitation with CSS does not allow the space of feasible utility parameters W to become empty (inconsistent) during the procedure.

In Section 3, we generalize this approach by generating optimal query *sets* (i.e., choice queries) that provide the greatest reduction of *a posteriori* regret. We consider both optimal computation as well as a generalization of the CSS strategy well-suited to query sets of any size.

2.5. Bayesian Recommendation and Adaptive Elicitation

We now outline the Bayesian framework. In this approach, the recommender's belief state θ is a probability distribution $P(w)$ over W, the space of utility parameters. In contrast to the regret-based approach, it has the advantage of being able to handle noisy responses and different forms of user-choice behavior more readily and can exploit probabilisitic prior information about user preferences. This comes at the expense of (typically) more complex belief state update.

2.5.1. Bayesian Recommendations

In the Bayesian approach, the recommender's uncertainty is captured by a distribution $P(w;\theta)$ over W, parameterized by θ [32, 18]. We will refer to both θ and $P(\cdot;\theta)$ as the recommender's belief state depending on context.

Definition 7 *Given distribution $P(\cdot;\theta)$, the* expected utility *of an item x is the expectation of $u(x;w)$ w.r.t. $w \sim P(\cdot;\theta)$:*

$$\mathrm{EU}(x;\theta) = \mathbb{E}_{P(w;\theta)}[u(x;w)] = \int_W u(x;w)P(w;\theta)dw. \tag{6}$$

If required to make a recommendation given belief θ, an optimal item $x^*(\theta)$ is one that has greatest expected utility.

Definition 8 *Given a distribution $P(w;\theta)$ and belief state θ, the* expected maximum utility *of θ and the* optimal item *are, respectively,*

$$\mathrm{EU}^*(\theta) = \max_{x\in X} \mathrm{EU}(x;\theta), \tag{7}$$

$$x^*(\theta) \in \operatorname*{argmax}_{x\in X} \mathrm{EU}(x;\theta). \tag{8}$$

This is the standard Bayesian decision criterion given utility function uncertainty.

While optimality is defined with respect to maximizing expected utility, we can equivalently minimize expected regret, which will sometimes be convenient when comparing it to the regret-based framework. Specifically, given the definition of regret $R(x;w)$ above, the expected regret of an item is simply $\mathbb{E}_{P(w;\theta)}R(x;w)$. It is easy to see that minimizing expected regret gives recommendations that maximize expected utility:

$$\arg\min_{x\in X} \mathbb{E}_{P(w;\theta)}R(x;w) = \arg\min_{x\in X} \int_W R(x;w)P(w;\theta)dw = \arg\max_{x\in X} \mathrm{EU}(x;\theta).$$

2.5.2. Bayesian Utility Elicitation

In the Bayesian framework, when a user responds to preference query $q \in Q$ with a response $r \in \mathcal{R}_q$, the recommender's belief state is updated using Bayes' rule to condition on that response. We denote the *updated belief state* by $\theta|r$. This posterior belief induces a new evaluation of items, $\mathrm{EU}(\cdot;\theta|r)$ with respect to $\theta|r$, and a (potentially) new optimal recommendation $x^*(\theta|r)$ with expected utility $\mathrm{EU}^*(\theta|r)$.

We assume a known *response model* $P(r|q,w)$ which dictates the probability of response $r \in \mathcal{R}_q$ if a user with (true) utility w is asked query q. We discuss response models in detail in Sections 4.3 and 4.4. To reduce notational clutter, we write $P(r|w)$, removing mention of q (e.g., which is valid if we assume, say, that $\mathcal{R}_q$ and $\mathcal{R}_{q'}$ are disjoint for any distinct queries q and q').

Conditioning on query responses poses some technical challenges since most common prior distributions are not conjugate with respect to typical preference-based evidence, in particular, responses to comparison and choice queries.

Several approximate inference methods can be used. For example, Bayesian updates can be performed using Monte Carlo methods, or inference schemes based on expectation-propagation [92], such as Trueskill [63]; these methods can be adapted to preference elicitation [59]. Our main contributions are independent of the method used for exact or approximate Bayesian inference.

A natural criterion for selecting queries in Bayesian approaches for elicitation is the maximization of (myopic) *expected value of information (EVOI).*

Definition 9 *Assume utility distribution $P(w;\theta)$, belief state θ, and response set $\mathcal{R}_q$ for some query q. The* expected posterior utility (EPU) *of q is*

$$\mathrm{EPU}(q;\theta) = \sum_{r\in\mathcal{R}_q} P(r;\theta)\ \mathrm{EU}^*(\theta|r). \tag{9}$$

where $P(r;\theta)$ is the probability of each response given belief θ:

$$P(r;\theta) = \int_W P(r|w)P(w;\theta)dw. \tag{10}$$

The (myopic) expected value of information *of q is*

$$\mathrm{EVOI}(q;\theta) = \mathrm{EPU}(q;\theta) - \mathrm{EU}^*(\theta). \tag{11}$$

Intuitively, the EVOI of q is the expected improvement in (expected) recommendation quality it offers relative to the current belief state θ, taking expectation over possible responses $r \in \mathcal{R}_q$.

This leads to the obvious definition of a (Bayesian) optimal query.

Definition 10 *Assume utility distribution P, belief state θ, and query set Q. The* optimal query *w.r.t. Q is that with greatest EVOI:*

$$q^*(\theta) = \arg\max_{q\in Q} \mathrm{EVOI}(q;\theta). \tag{12}$$

We note that *sequential* EVOI is, in general, a more effective elicitation criterion if the goal is to quickly improve recommendation quality, since it allows consideration of the cumulative impact of multiple elicitation queries. The problem of deciding which questions to ask in this case can be formulated as a partially observable Markov decision process [18, 66]. However, such a model is impractical to solve for nontrivial problems. Indeed, myopic EVOI is much more commonly used in decision analysis than sequential EVOI for this reason [67]. However, even the computation and optimization of myopic EVOI for online query selection is often computationally prohibitive; hence, in practice, heuristics are often used that offer no theoretical guarantees with respect to query quality [69, 50].

In Section 4, we develop the Bayesian approach to optimization and elicitation for generating optimal recommendation sets with greatest expected utility and optimal choice queries that have the greatest EVOI. In particular, we show that myopic EVOI can be computed relatively effectively for choice queries.

2.6. Related Approaches

The study of the assessment of the preferences of a decision maker goes back several decades. Particular emphasis has been given to the elicitation of utility functions for multi-attribute and multi-criteria settings [74]. Classic approaches to utility elicitation often focus on high-stakes decision scenarios and aim to assess the decision maker's utility very precisely. The decision maker is asked a number of questions to determine the parameters of the utility function, with the exact questions depending on the adopted protocol.

While classic elicitation protocols are well-founded, their applicability to many practical recommendation settings is limited, as they suffer from a number of drawbacks. For instance, classic *standard gamble queries* (and similar questions) are difficult for users to answer. The precision attained with classical elicitation methods is often unjustified, since the cognitive cost imposed on a decision maker might not be worth the increase in decision quality. Some work [71, 7] addresses elicitation of a utility function when only incomplete information is available. Instead of fully eliciting an utility function, indirect elicitation methods assess the utility function from assignment examples (for instance, examples are assigned to classes).

The UTA method [71] is an assessment procedure for a set of utility functions based on linear programming. Local utilities are piecewise linear; the interval is divided in subintervals and linear interpolation is used to approximate the utility contribution of a given feature. Pairwise preferences are expressed with linear inequalities; slack variables are added to allow inconsistencies in the preference information. Since several utility functions may be feasible, one typically minimizes the sum of the slack variables, yielding an utility function that fits the available preference information as well as possible (giving somewhat robust recommendations). An alternative framework is MACBETH [7], also based on a system of linear inequalities, that asks the user to provides reference levels for each feature/criterion and information about the difference in satisfaction between the values of a given feature. Utility elicitation methods go beyond additive models—several researchers have considered models based on the Choquet integral, e.g., the TOMASO decision support system [89] and the MYRIAD software tool set [81].

The fact that generally many user utility functions are consistent with partial preference information gives rise to robustness concerns. Assume that at a given point in the elicitation process that W is the set of possible utility functions. In *robust ordinal regression* [58], a choice x is necessarily preferred to y, denoted $x \succeq^N y$, if, for all $w \in W$, $u(y;w) \geq u(x;w)$ (with strict necessary preference $x \succ^N y$ if $u(y;w) > u(x;w)$ for all $w \in W$); x is possibly preferred to y, written $\succeq^P$, if there exists at least one utility function such that $u(x;w) \geq u(y;w)$ with $w \in W$. The method UTA^{GMS} provides linear programming formulations in order to compute necessary and possible rankings consistent with an underlying utility. When additional preference statements are added, the relation $\succeq^P$ can only decrease and $\succeq^N$ increase.

Notice that the minimax regret implicitly subsumes the notion of possible and necessary preferences:

- If $\text{PMR}(y,x;W) \geq 0$ then x is possibly (weakly) preferred to y, i.e., $x \succeq^P y$; moreover, we can distinguish $\text{PMR}(y,x;W) > 0$, where x is possibly strictly preferred to y, and $\text{PMR}(y,x;W) = 0$, where x and y are possibly equally preferred, $x \sim^P y$.
- If $\text{PMR}(x,y;W) < 0$ then x is necessarily strictly preferred to y ($x \succ^N y$). If $\text{PMR}(x,y;W) = 0$ then x is necessarily weakly preferred to y ($x \succeq^N y$).
- If $\text{PMR}(x,y;W) = \text{PMR}(y,x;W) = 0$ then x is necessarily equally preferred to y: $x \sim^N y$.

The *analytical hierarchical process (AHP)* [108] is a widely used method for decision support in which the decision maker provides pairwise information about the intensity of the different criteria, which are mapped to a numerical scale, determining an evaluation matrix. A weighting vector is then computed from the principal eigenvector of the matrix. From pairwise preference statements about the items according to a specific criterion, a general ranking is obtained. Despite its popularity, AHP has been criticized for several reasons, including the number of queries that the typical decision maker must answer in complex scenarios (for example, if there are 20 different models of, say, cameras and 10 relevant criteria, the user will be asked 520 questions, which would be unacceptable in typical electronic commerce applications).

3. Setwise Recommendation and Elicitation under Minimax Regret

In this section, we develop setwise recommendation and utility elicitation within the regret-based framework outlined in Section 2.4. Our contributions are organized as follows. In Section 3.1, we extend the minimax regret criterion to recommendation sets of arbitrary size. In Section 3.2, we formalize the notion of worst-case *a posteriori* regret (WR) for queries to the case of choice queries, i.e., when treating a set S of arbitrary size as a query set as opposed to a recommendation set. (Note that the case of sets of size two is the standard comparison query addressed previously in the literature.) In Section 3.3, we establish the theoretical connection between optimal recommendation sets and (myopically) optimal query sets, showing that they coincide. After an illustration of these notions with a detailed example in Section 3.4, we devise several exact and heuristic algorithms for computing of sets of items that serve the dual purpose of recommendation and elicitation in Section 3.5.

3.1. Recommendation Sets

As discussed in Section 2, when the recommender is uncertain about a user's true utility function, there can be value in recommending a set of options to the user assuming she will select her most preferred from that set. We now generalize minimax regret to cover the case of such *recommendation sets* under strict uncertainty.

Let W denote the set of feasible utility parameters, reflecting any prior preference information elicited from the user, such that the user's utility function is some $w \in W$. We do not consider noisy or erroneous information in this section. Our aim is to recommend an item set $S \subseteq X$, assuming the user selects her most preferred from S, that is robust to the uncertainty/imprecise knowledge inherent in W. For concreteness, we assume that the maximum cardinality of S is fixed to be $k < |X|$.[3]

We now quantify the potential loss of restricting the user's decision to items in some slate S (which we assume to be of size k). Intuitively, the user may select any of the k options as being the most preferred. An adversary wanting to maximize regret (or user loss) should do so assuming that any such choice is possible—unlike single-item max regret, we allow the user to select *any* item from S. Informally, we choose the set of k items first, but "postpone" the user's choice from S until *after* the adversary has chosen w. The regret of S is the difference between the utility of the best configuration under w and the utility of the best option w.r.t. w in S.

Definition 11 *Assume feasible utility space W. The* setwise max regret *of a recommendation set S and the corresponding* setwise adversarial choice *are:*

$$\mathrm{SMR}(S;W) = \max_{y\in X}\max_{w\in W}\left\{u(y;w) - \max_{x\in S} u(x;w)\right\}, \tag{13}$$

$$\mathit{SMR\text{-}AC}(S;W) \in \operatorname{argmax}_{y\in X}\max_{w\in W}\left\{u(y;w) - \max_{x\in S} u(x;w)\right\}. \tag{14}$$

The *setwise max regret (SMR)* of set S reflects the intuitions above, and *SMR-AC*(S,W) is the adversarial alternative that maximizes regret (i.e., an item in X that under the adversarially selected utility function w would have given the user greatest utility).

Given w, the option S that determines SMR is that with highest utility with respect to w. Intuitively, SMR represents the worst-case loss of requiring the user adopt some item $x \in S$ when the user's optimal item instead lies in $X \setminus S$.

Under this performance criterion, the recommender should offer a recommendation set with minimum SMR.

Definition 12 *Assume feasible utility space W. The* setwise minimax regret *and the* minimax optimal set *w.r.t. W are:*

$$\mathrm{SMMR}(W) = \min_{S\subseteq X;\,|S|=k} \mathrm{SMR}(S;W) = \min_{S\subseteq X;\,|S|=k}\max_{y\in X}\max_{w\in W}\left\{u(y;w) - \max_{x\in S} u(x;w)\right\}, \tag{15}$$

$$S^*_W \in \arg\min_{S\subseteq X;\,|S|=k} \mathrm{SMR}(S;W) = \arg\min_{S\subseteq X;\,|S|=k}\max_{y\in X}\max_{w\in W}\left\{u(y;w) - \max_{x\in S} u(x;w)\right\}. \tag{16}$$

Setwise minimax regret (SMMR) is the SMR of the minimax optimal set S^*_W, that is, the set that minimizes SMR(S,W). We sometimes refer to S^*_W simply as an optimal recommendation set when it is clear from context that we are referring to the SMMR criterion.

We outline a few important properties of SMR and SMMR (proofs are omitted in cases that are easily verified). First, SMR is *monotonic*—adding items to a recommendation set cannot increase SMR (and removing items cannot decrease SMR):

[3] We can interpret k as an upper bound on the size of the recommendation set. We will see that, under the hypothesis that the user is able to correctly identify his most preferred item, there is never a reason to recommend fewer than k, unless there is some cost associated with larger slate (e.g., attentional cost). When dealing with the Bayesian framework, we consider *noisy* response models where the quality of a recommendation set is not monotone.

Observation 2 $\mathrm{SMR}(A \cup B; W) \leq \mathrm{SMR}(A; W)$ *for all* $A, B \subseteq X$.

Monotonicity justifies our focus on recommendation sets that contain exactly k items, as any subset containing fewer than k items cannot have lower SMR than the best of size k.

It is easy to show that incorporating options that are known to be dominated given W does not change SMR. When considering a set of items, the following notion of dominance is pertinent. An item a is *setwise dominated* in S if, for every utility function $w \in W$, there is an item $b \in S$ such that b has higher utility than a.

Observation 3 *Assume feasible utility space* W*, a recommendation set* S *and an item* b*. If for all* $w \in W$ *there is some* $a \in S$ *such that* $u(a; w) \geq u(b; w)$*, then* $\mathrm{SMR}(S \cup \{b\};\ W) = \mathrm{SMR}(S; W)$.

There are several alternative ways of rewriting SMR, for example:

Observation 4 *Setwise max regret can be formulated as:*

$$\mathrm{SMR}(S; W) = \max_{y \in X} \max_{w \in W} \min_{x \in S} \quad u(y; w) - u(x; w), \tag{17}$$

$$\mathit{SMR\text{-}AC}(S; W) \in \arg\max_{y \in X} \max_{w \in W} \min_{x \in S} \quad u(y; w) - u(x; w). \tag{18}$$

This observation captures the intuition that, given a utility function w, SMR is computed by considering, for each w, the option (among those in S) with minimum loss against the adversarial choice y; the adversary will then choose w and y to maximize this loss.

It is useful to reason with the S-cover of W (see Definition 1). An important observation that we use later is that $\mathrm{SMR}(S, W)$ can be determined using the maximum of the item-wise max-regret values over the S-cover:

Proposition 5 *Let* $S = \{x_1, \ldots, x_k\}$ *and let* $\{W_{S \trianglerighteq x_i}\}_{i \in I}$*, with* $I \subseteq \{1, \ldots, k\}$ *be an* S-cover *of* W*. Then*

$$\mathrm{SMR}(S; W) = \max\left\{ \mathrm{MR}(x_i; W_{S \trianglerighteq x_i}) : i \in I \right\}.$$

Proof.

$$\begin{aligned}\mathrm{SMR}(S; W) &= \max_{y \in X} \max_{w \in W} \left\{ u(y; w) - \max_{x \in S} u(x; w) \right\} = \max_{y \in X} \max_{x \in S} \max_{w \in W_{S \triangleright x}} \left\{ u(y; w) - \max_{x \in S} u(x; w) \right\} \\ &= \max_{x \in S} \max_{w \in W_{S \triangleright x}} \max_{y \in X} \left\{ u(y; w) - u(x; w) \right\} = \max_{x \in S} \max_{w \in W_{S \triangleright x}} R(x; w) = \max\left\{ \mathrm{MR}(x_i; W_{S \triangleright x_i}) : 1 \leq i \leq k \right\}.\end{aligned}$$

□

This observation is quite useful in practice: it implies that SMR can be calculated using k standard (item-wise) max regret computations.

3.2. Elicitation with Choice Queries

Sets of items are used not just to make a recommendations, but also to give the user the opportunity to critique the proposed options and continue exploration of the item space. Our setwise minimax regret criterion can be used directly to generate sets that serve as such choice queries, and in fact implements a form of *preference-based diversity*. This stands in contrast to "product diversity" typically considered in critiquing systems. Furthermore, unlike work in polyhedral conjoint analysis [126], which emphasizes volume reduction of the utility polytope W, our regret-based criterion is sensitive to the range of feasible items and does not reduce utility uncertainty for its own sake.

Any set S can be interpreted as a query by simply allowing the user to state which of the k elements $x_i \in S$ she prefers. We refer to S interchangeably as a *choice query* or a *choice set* when used in this fashion. The user's identification of some $x_i \in S$ as her most preferred (with ties broken in an arbitrary way) refines the set of feasible utility functions W by imposing the following $k-1$ constraints on W:

$$u(x_i; w) \geq u(x_j; w), j \neq i.$$

When treating S as a choice query, we are not interested in its max regret (as we would if it were a recommendation set), but rather in *how much a response to that query might reduce minimax regret.*

Intuitively, each possible response x_i to the choice query S gives rise to updated beliefs about the user's utility function. If all answers are possible—that is, if, for all $x_i \in S$, there is some $w \in W$ under which x_i has maximum utility among items in S—the set of responses associated with a query S yields a vector of minimax regret values

$$(\text{MMR}(W_{S \triangleright x_1}), \ldots, \text{MMR}(W_{S \triangleright x_k}))$$

in each of the updated feasible spaces $W_{S \triangleright x_1}, ..., W_{S \triangleright x_k}$. More generally, assume $\{W_{S \trianglerighteq x_i}\}_{i \leq k}$ is the S-cover of the set S. Then the vector of minimax regret values, considering only the responses x_i that are consistent with W, is given by $(\text{MMR}(W_{S \triangleright x_i}))_{i \leq k}$.

Myopic worst-case regret WR (Definition 6) is a measure of the value of information of a query in the regret-based, distribution-free setting. For choice queries S, WR simply requires that we consider the greatest minimax regret value over the updated spaces, with the most valuable choice query being that which minimizes WR.

Definition 13 *Let W denote the feasible utility space and assume $\{W_{S \trianglerighteq x_i}\}_{i \leq k}$ is the S-cover of the set S. The (myopic)* worst-case a posteriori regret (WR) *of choice query $S = \{x_1, \ldots, x_k\}$ is*

$$\text{WR}(S; W) = \max \left\{ \text{MMR}(W_{S \triangleright x_i}) \right\}_{i \leq k}. \tag{19}$$

An optimal query set *is any set S^*_W that has minimum worst-case regret* $\text{WR}^*(W)$*:*

$$\text{WR}^*(W) = \min_{S \subseteq X;\, |S|=k} \text{WR}(S; W) \tag{20}$$

$$S^*_W \in \arg \min_{S \subseteq X;\, |S|=k} \text{WR}(S; W). \tag{21}$$

We use the worst-case response to measure the quality of the query (i.e., the response that leads to the updated W with greatest remaining minimax regret); the optimal query is that which minimizes this value. The worst-case improvement of minimax regret associated with S is $\text{MMR}(W) - \text{WR}(S; W)$. When considering the query that gives the greatest worst-case improvement in minimax regret (in the worst-case), it is sufficient to minimize WR, as the current minimax regret value MMR is the same for all candidate queries.

3.3. The Connection between SMR *and* WR

We analyze the connection between the value of a recommendation set in terms of regret (measured by setwise max regret SMR) and its value as a query set (measured by worst-case regret WR).

First of all, it is not hard to show that the worst-case regret of choice set S is never greater than its setwise max regret:

Proposition 6 $\text{WR}(S; W) \leq \text{SMR}(S; W)$.

Proof. This follows directly by considering Proposition 5, Definition 13 and the fact that $\text{MMR}(W_{S \triangleright x_i}) \leq \text{MR}(x_i, W_{S \triangleright x_i})$ for any $i \in I$ (by definition, minimax regret cannot greater than max regret of some item in the same utility subspace), where I is the set items whose choice is consistent:

$$\text{WR}(S; W) = \max \left\{ \text{MMR}(W_{S \triangleright x_i}) \right\}_{i \in I} \leq \max \left\{ \text{MR}(x_i, W_{S \triangleright x_i}) \right\}_{i \in I} = \text{SMR}(S; W).$$

□

In the sequel we make use of a transformation T that modifies a given recommendation set S in such a way that SMR cannot increase, and usually decreases. This transformation will be used in two ways: to prove the optimality of SMR for choice set generation; and directly as a computationally viable heuristic strategy for choice set generation. The operator T refines a recommendation set S as follows:

(a) first construct an S-cover of W;

(b) then compute a *single recommendation* (Definition 5) that has minimax regret in each element in the partition (i.e., each induced region of utility space); and

(c) let $T(S)$ be the new recommendation set consisting of these new recommendations.

As there might be more than one option that achieves minimax regret in each element of the partition, we assume that $T(S)$ is obtained by breaking ties in some (potentially) non-deterministic way. We use $\tau(S)$ to refer to the set of all possible sets $T(S)$ that could be constructed in this way. We define T and τ more formally as follows:

Definition 14 *Let $S = \{x_1, \dots, x_k\}$ and its S-cover be $\{W_{S \triangleright x_i}\}_{i \leq k}$. Define the T operator as:*

$$T(S) = \{x^*_{W_{S \triangleright x_i}}\}_{i \leq k}.$$

Define $\tau(S)$ as the collection of sets $\{x_i\}_{i \leq k}$ where each x_i has minimax regret in the corresponding element of the S-cover $W_{S \triangleright x_i}$:

$$\tau(S) = \{\{x_i\}_{i \leq k} | x_i \in \arg\min \text{MR}(x; W_{S \triangleright x_i})\}.$$

Note that $T(S)$ may have a smaller cardinality than S. Whenever $S \in \tau(S)$, we say that S is a fixed point of T.

The optimality of minimax-optimal setwise recommendations, when used as choice queries, is based on the following lemma, asserting that the set resulting from the application of T has setwise regret less than the worst-case regret of the original set. In other words, for any set S, $T(S)$ is at least as good as S—both as recommendation set and as a choice query—and, more precisely, the value of $T(S)$ as a recommendation set is at least that of set S as query.

Lemma 7 (*Regret-based Query Iteration*) SMR$(T(S); W) \leq$ WR$(S; W)$.

Proof. Let $S = \{x_1, \dots, x_k\}$ and $T(S) = \{x'_1, \dots, x'_k\}$, where $x'_i = x^*_{W_{S \triangleright x_i}}$. Assume $\{S \triangleright x_i\}_{i \leq k}$ is the S-cover and $\{T(S) \triangleright x_j\}_{j \leq k}$ is the $T(S)$-cover. For each $x_i \in S$, $x'_j \in T(S)$, consider $W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}$, the subset of W where x_i is preferred in choice set S and x'_j is preferred in choice set $T(S)$. These subsets form the "meet" cover obtained by intersecting each element of the S-cover and of the $T(S)$-cover: the set of intersections $\{W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}\}_{i,j \leq k}$ has at least one non-empty element and covers W; we ignore empty intersections. We now formulate WR$(S; W)$ and SMR$(T(S); W)$ as follows:

$$\text{WR}(S; W) = \max_{i \in I} \left\{ \text{MR}\left(x'_i; W_{S \triangleright x_i}\right) \right\} = \max_{i,j \leq k: W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j} \neq \emptyset} \left\{ \text{MR}\left(x'_i; W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}\right) \right\}, \tag{22}$$

$$\text{SMR}(T(S); W) = \max_{j \leq k} \left\{ \text{MR}\left(x'_j; W_{T(S) \triangleright x'_j}\right) \right\} = \max_{i,j \leq k: W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j} \neq \emptyset} \left\{ \text{MR}\left(x'_j; W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}\right) \right\}. \tag{23}$$

Equation 22 uses first Definition 13 and the fact x'_i is minimax regret optimal in $W_{S \triangleright x_i}$, and then further divides utility space with respect to $T(S)$-cover. Similarly, Equation 23 uses Proposition 5 and then further divides utility space according to the S-cover. We now compare the components of Equation 22 and Equation 23 with respect to utility space $W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}$, for each $i \leq k$ and $j \leq k$ such that the intersection is not empty:

- if $i = j$ then the two MR components are the same;
- if $i \neq j$, consider any $w \in W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}$. Since $w \in W_{T(S) \triangleright x'_j}$, we must have $u(x'_j; w) \geq u(x'_i; w)$. Therefore MR$(x'_j, W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j}) \leq$ MR$(x'_i, W_{S \triangleright x_i} \cap W_{T(S) \triangleright x'_j})$.

In the expression of $\mathrm{SMR}(T(S); W)$ (see Eq. 23), each component is no greater than its correspondent in the $\mathrm{WR}(S)$ expression (see Eq. 22). Thus $\mathrm{SMR}(T(S); W) \leq \mathrm{WR}(S; W)$. □

From Lemma 7 and Proposition 6, it follows that T cannot increase setwise max regret:

$$\mathrm{SMR}(T(S); W) \leq \mathrm{SMR}(S; W)$$

We will exploit this observation algorithmically in the local search strategy we describe in Section 3.5.

These observations lead to our main result connecting choice queries and recommendations sets in the regret-based framework:

Theorem 8 *Let S^*_W be an optimal recommendation set of size k, $S^*_W = \arg\min_{S\subseteq X;\, |S|=k} \mathrm{SMR}(S, W)$. Then:*

- *S^*_W is an optimal choice query: $S^*_W \in \arg\min \mathrm{WR}(S; W)$*
- *the value of the optimal set as a recommendation coincides with its value as query: $\mathrm{SMR}(S^*_W; W) = \mathrm{WR}(S^*_W; W)$, or, equivalently $\mathrm{SMMR}(W) = \mathrm{WR}^*(W)$.*

Proof. Suppose S^*_W is not an optimal choice set, i.e., there is some S' such that $\mathrm{WR}(S'; W) < WR(S^*_W; W)$. If we apply transformation T to S' we obtain a set $T(S')$, and by Lemma 7 we have: $\mathrm{SMR}(T(S'); W) \leq \mathrm{WR}(S'; W) < \mathrm{WR}(S^*_W; W) \leq \mathrm{SMR}(S^*_W; W)$. This contradicts the (setwise) minimax regret optimality of S^*_W.
To prove the second part, note that since S^*_W is an optimal recommendation set, $\mathrm{SMR}(S^*_W; W) \leq \mathrm{SMR}(T(S^*_W); W)$ holds. By Lemma 7, $\mathrm{SMR}(T(S^*_W); W) \leq \mathrm{WR}(S^*_W; W)$, and therefore $\mathrm{SMR}(S^*_W; W) \leq \mathrm{WR}(S^*_W; W)$. But since we have $\mathrm{WR}(S^*_W; W) \leq \mathrm{SMR}(S^*_W; W)$ by Proposition 6, we conclude that $\mathrm{SMR}(S^*_W; W) = \mathrm{WR}(S^*_W; W)$. □

The implications of this result are significant—in the regret-based framework where recommender uncertainty about a user's utility function is modeled in a strict (unquantified) fashion, the recommender need not tradeoff exploration and exploitation. The optimal recommendation set from which a user can select her "final" recommendation is also an optimal choice query set, in which the user can simply express her preference and the recommender can proceed directly to another round of elicitation. As a by-product, we also have that the optimal values of setwise max regret and worst-case *a posteriori* regret coincide.

3.4. Illustration

Example 1 *We illustrate the notions of setwise regret, optimal recommendation sets, and optimal choice queries with a very simple example, involving a domain with five items $x_1, \ldots, x_5$ defined over two features $x_i[1]$ and $x_i[2]$ as follows:*

	$x_i[1]$	$x_i[2]$
x_1	*0.35*	*0.68*
x_2	*0.9*	*0.2*
x_3	*0*	*0.75*
x_4	*1*	*0*
x_5	*0.5*	*0.3*

We use square brackets to denote a feature of an item, e.g., $x_1[2]$ denotes the second feature of item x_1. We assume linear user utility $u(x; w_1) = w_1 \cdot x[1] + (1 - w_1) \cdot x[2]$, with a single (unknown) parameter w_1, $0 \leq w_1 \leq 1$ and the weight of the second feature $w_2 = 1 - w_1$. For example, if $w_1 = 0.5$ then utility of x_1 is $u(x_1; w_1) = 0.35 \cdot 0.5 + 0.68 \cdot 0.5 = 0.51$.

This simple example is convenient because utility is one-dimensional and it is easy to visualize item utilities as a function of w_1: the utility of each x_i as a function of w_1 is shown in Figure 2. We notice that, for some values of w_1, each of the options x_1, x_2, x_3 and x_4 is optimal, but not so for x_5. However, x_5 is minimax optimal: its max regret of 0.5 (attained with adversarial choice x_4 at $w_1 = 1$) is less than that of all other options. This can be seen using a

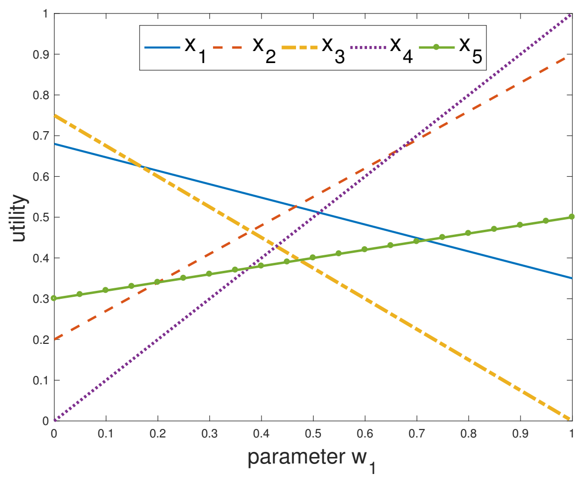


Figure 2: The utility of x_1, x_2, x_3, x_4, x_5 as a function of weight w_1 (see Example 1).

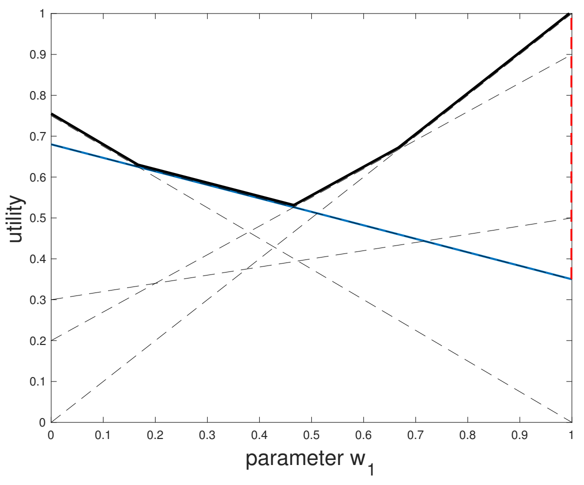


Figure 3: Max regret of x_1 is attained at $w_1 = 1$; it corresponds to the maximum difference between its utility and the black upper piecewise-linear function.

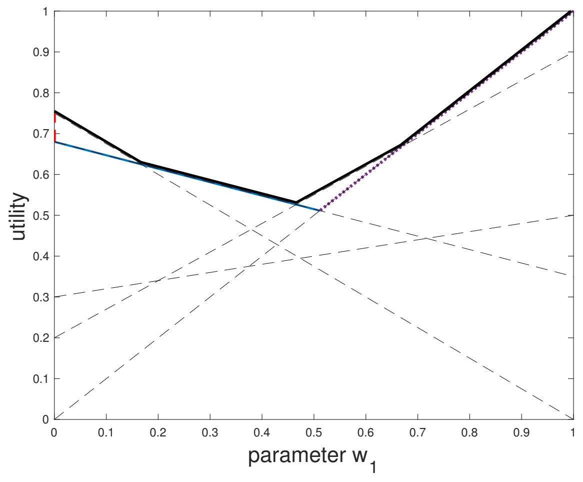


Figure 4: Setwise max regret of $\{x_1, x_4\}$ is attained with $w_1 = 0$; this pair achieves setwise minimax regret, and is the optimal recommendation set of size $k = 2$.

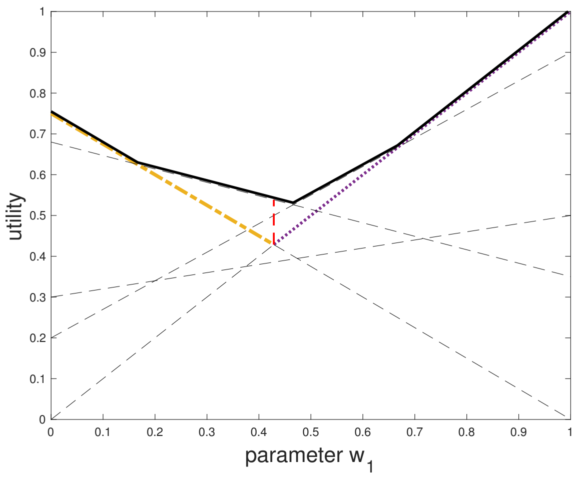


Figure 5: Setwise max regret of $\{x_3, x_4\}$ is attained with $w_1 = 0.4286$ (the value that makes x_3 and x_4 equally preferred).

geometric interpretation of max regret: x_5's utility is linear in w_1 and its max regret is its maximum distance from the piecewise linear, convex function of w_1 determined by the upper surface $\max_i u(x_i; w_1)$ (shown in black in Figure 3). This distance is less than that of any other item (e.g., the max regret of x_1 is 0.65, occurring at $w_1 = 1$, as shown in Figure 3, while the max regret x_2 is 0.55, occurring at $w_1 = 0$).

When considering a particular value of w_1, the true regret is the difference between the utility of the best option given w_1 and the utility of our recommendation. For instance, if $w_1 = 0.9$, the best item is x_4 with utility 0.9, while x_1 has utility 0.38; hence, $R(x_1; w_1) = 0.52$. Max regret accounts for the uncertainty over w_1 and is the maximum of the possible actual regret values (for x_1 it is 0.65 when $w_1 = 1$).

When the options are few as in this case, we can compute the max regret of each choice by explicitly enumerating its pairwise max regret w.r.t. any possible adversarial item choice. The table below illustrates this, where each row is a possible recommendation, each column is an adversarial choice, and PMR (allowing the adversary to choose w_1) is displayed in the cells. The max regret of each recommendation—its maximum PMR—is shown in the last column. With no preference information (e.g., prior to utility elicitation), w_1 can take any value between 0 and 1, and we have:

		Adversary		
Set	SMR	y	w_1	WR
$k = 2$				
$\{x_1, x_2\}$	0.1	x_4	1	**0.07**
$\{x_1, x_3\}$	0.65	x_4	1	0.308
$\{x_1, x_4\}$	**0.07**	x_3	0	**0.07**
$\{x_1, x_5\}$	0.5	x_4	1	0.273
$\{x_2, x_3\}$	0.1	x_4	1	0.1
$\{x_2, x_4\}$	0.55	x_3	0	0.206
$\{x_2, x_5\}$	0.45	x_3	0	0.274
$\{x_3, x_4\}$	0.11	x_1	0.4286	0.1
$\{x_3, x_5\}$	0.5	x_4	1	0.07
$\{x_4, x_5\}$	0.45	x_3	0	0.1

		Adversary		
Set	SMR	y	w_1	WR
$k = 3$				
$\{x_1, x_2, x_3\}$	0.100	x_4	1	0.06
$\{x_1, x_2, x_4\}$	0.070	x_3	0	0.07
$\{x_1, x_2, x_5\}$	0.100	x_4	1	0.07
$\{x_1, x_3, x_4\}$	**0.047**	x_2	0.5113	**0.047**
$\{x_1, x_3, x_5\}$	0.500	x_4	1	0.273
$\{x_1, x_4, x_5\}$	0.070	x_3	0	0.07
$\{x_2, x_3, x_4\}$	0.089	x_1	0.3793	0.089
$\{x_2, x_3, x_5\}$	0.100	x_4	1	0.1
$\{x_2, x_4, x_5\}$	0.450	x_3	0	0.206
$\{x_3, x_4, x_5\}$	0.110	x_1	0.4286	0.1
$k = 4$				
$\{x_1, x_2, x_3, x_4\}$	**0**	x_1	0.1667	**0**
$\{x_1, x_2, x_3, x_5\}$	0.1	x_4	1	0.06
$\{x_1, x_2, x_4, x_5\}$	0.07	x_3	0	0.07
$\{x_1, x_3, x_4, x_5\}$	0.047	x_2	0.5113	0.047
$\{x_2, x_3, x_4, x_5\}$	0.089	x_1	0.3793	0.089

Table 1: Setwise max regret values for $k = 2, 3, 4$ (see Example 1).

PMR(x_i, x_j)	x_j					
x_i	x_1	x_2	x_3	x_4	x_5	MR(x_i)
x_1	0	0.55	0.07	0.65	0.15	0.65
x_2	0.48	0	0.55	0.1	0.1	0.55
x_3	0.35	0.9	0	1.00	0.5	1
x_4	0.68	0.2	0.75	0	0.3	0.75
x_5	0.38	0.4	0.45	0.5	0	0.5

Minimax regret is 0.5, and the minimax optimal recommendation is option x_5; its max regret occurs at adversarial choice of utility $w_1 = 1$, and the adversarial choice is x_4. It can be easily shown that regret is maximized at one of the vertices of the feasible region W when it is a bounded, convex polytope (such as the polytope induced by responses to choice queries).

Recommendation sets have a similar graphical interpretation: instead of considering a single hyperplane, we consider the upper surface dictated by the recommendation set (i.e., the piecewise-linear, convex maximum function) and its distance from the full upper surface. For instance, consider the recommendation of $k = 2$ items, $\{x_1, x_4\}$. By Proposition 5 we partition utility space at the point where the two items have identical utility, $w_1 = 0.5113$, and use x_1 for $w_1 < 0.5113$ and x_4 for $w_1 \geq 0.5113$. The setwise max regret is the maximum difference between the maximum of these two item utilities (as a function of w_1) and the upper surface: SMR$(\{x_1, x_4\})$ is 0.07 and occurs at $w_1 = 0$ (see Figure 4).

Table 1 shows how recommendation sets of cardinality 2, 3, 4 are ranked w.r.t. SMR and WR. The pair $\{x_1, x_4\}$ is in

Set S	$\{x_1, x_2, x_3\}$	$\{x_1, x_2, x_4\}$	$\{x_1, x_2, x_5\}$	$\{x_1, x_3, x_4\}$	$\{x_1, x_3, x_5\}$
$T(S)$	$\{x_1, x_3, x_4\}$	$\{x_1, x_2, x_4\}$	$\{x_1, x_4\}$	$\{x_1, x_3, x_4\}$	$\{x_1, x_3, x_4\}$
Set S	$\{x_1, x_4, x_5\}$	$\{x_2, x_3, x_4\}$	$\{x_2, x_3, x_5\}$	$\{x_2, x_4, x_5\}$	$\{x_3, x_4, x_5\}$
$T(S)$	$\{x_1, x_4\}$	$\{x_1, x_2, x_4\}$	$\{x_1, x_4\}$	$\{x_1, x_3, x_4\}$	$\{x_1, x_4\}$

Table 2: Application of the T operator to sets of size three (Example 1).

fact setwise minimax optimal among all 2-sets, though other pairs, e.g., $\{x_1, x_2\}$ have relatively low SMR. Notice that any pair containing x_5, the best singleton recommendation, is in fact a *poor* set recommendation and has relatively high SMR. Moreover, we see that an optimal set recommendation can often have dramatically lower max regret than the minimax optimal *single* recommendation (in this case, x_5). Indeed, the optimal recommendation set may consist of options (in this case, x_1 and x_4) that, when considered alone, have high max regret. As can be seen in the table, the adversary's utility does not necessarily correspond to a vertex of the feasible region W, as in the single recommendation case; it may also lie in any intersection of the hyperplanes associated with the options. For instance the setwise regret of $\{x_3, x_4\}$ is maximized at $w_1 = 0.4286$, the utility function that makes x_3 and x_4 equally preferred (see Figure 5).

If we consider recommendation sets with $k = 3$, SMMR is an even lower 0.047, with optimal set $\{x_1, x_3, x_4\}$ (adversarial choice x_2; $w_1 = 0.5113$, at which x_1 and x_4 yield the same utility). The optimal four-item set $\{x_1, x_2, x_3, x_4\}$ has SMR 0: the slate includes all the options that the user could consider optimal. (By Proposition 5, SMR must be 0 since for any $W_{S \triangleright x_i}$ that partitions W, x_i is optimal in $W_{S \triangleright x_i}$).

Table 1 also displays the values of WR (the minimax regret "a posteriori") for the sets, measuring the value of using the set as a choice query. A small value of WR means that the choice query is very informative; these values can be compared to the current minimax regret value (0.5) to measure how much MMR is reduced in the worst case (considering all possible answers). By comparing the values of SMR and WR for each set, we can verify Proposition 6 for this instance, which ensures that, for any set, WR is at most its SMR value. As guaranteed by Theorem 8, for a given k, the optimal recommendation set is also an optimal query, and indeed the optimal SMR value coincides with the optimal WR value. For $k = 2$, the optimal recommendation set $\{x_1, x_4\}$ is also an optimal query set. Notice, however, that $\{x_1, x_2\}$ is also an optimal query set without being an optimal recommendation set.

We now turn our attention to the operator T (Definition 14); applying T to any pair returns the optimal 2-set $\{x_1, x_4\}$, which is the only fixed point for $k = 2$. We list the results of applying T to any triplet in Table 2.

As we can see, in a few cases T returns a set of lower cardinality. For instance, with $\{x_3, x_4, x_5\}$: for any value of w_1, either x_3 or x_4 has higher utility than x_5, so the space $W_{\{x_3, x_4, x_5\} \triangleright x_5}$ is empty. Note that, when starting with a set of size three, there are three different fixed points for T: $\{x_1, x_2, x_4\}$, the optimal set $\{x_1, x_3, x_4\}$, and the pair $\{x_1, x_4\}$.

3.5. Algorithms

We now derive several algorithms to generate good sets of items to be used as set recommendations, choice queries, or both. In Section 3.5.1 we describe how to compute the optimal set with respect to SMR, and in Section 3.5.2 we develop faster heuristic methods.

3.5.1. Computation of the Optimal Set

We describe how to compute regret-based recommendations, distinguishing *configuration problems*—in which items are specified as instantiations of variables to be configured subject to various feasibility constraints, generally as part of a combinatorial optimization process—from *database problems*—in which items are specified explicitly in a "tabular" fashion.

Configuration Problems. In configuration problems, options are defined by a set of variables and configuration constraints (i.e., as solutions to a constraint satisfaction problem). In such domains, minimax regret computation for a *single* recommendation can be formulated as a mixed-integer program (MIP), and solved practically for large problems using techniques such as Bender's decomposition and constraint generation. We refer to [21, 25, 19] for more details. Our MIP formulations for setwise minimax regret below draw heavily on these techniques, but require important modifications.

We present here the MIP formulation of *setwise* minimax regret for configuration problems in the case of linear utility functions $u(x; w) = w \cdot x$. We assume that the *feasibility* constraints (encoding which configurations are valid) are linear and write X_{feasible} to denote the set of feasible configurations.

$$
\begin{aligned}
&\min \quad M && && (24)\\
&\text{s.t. } M \geq \sum_{1\leq j\leq k} R^j_w && \forall w \in W && (25)\\
&\quad R^j_w \geq w(x^*_w - X^j) + (I^j_w - 1)a && \forall j \in \{1,\dots,k\}, w \in W && (26)\\
&\quad \sum_{1\leq j\leq k} I^j_w = 1 && \forall w \in W && (27)\\
&\quad M \geq 0 && && (28)\\
&\quad I^j_w \in \{0,1\} && \forall j \in \{1,\dots,k\}, w \in W && (29)\\
&\quad R^j_w \geq 0 && \forall j \in \{1,\dots,k\}, w \in W && (30)\\
&\quad X^j \in X_{\text{feasible}} && \forall j \in \{1,\dots,k\} && (31)
\end{aligned}
$$

Decision variables are represented by uppercase letters (M, X, R, I). Intuitively, this MIP chooses k items $x_j, j \leq k$, designated by variables X^j (where each X^j is a variable vector over the n item attributes). Each X^j must be feasible, hence belong to X_{feasible}. The objective minimizes M subject to constraint (25), ensuring M is no less than the setwise regret of the selected options w.r.t. any $w \in W$ (i.e., no less than setwise max regret). At the optimum, the value of M is the setwise minimax regret.

Here R^j_w is the true regret of jth option X^j w.r.t. w when the indicator variable I^j_w is active (indicating that the jth option is the item in the set with maximal utility under w). For any w, exactly one I^j_w is set to 1. Hence minimization of M ensures $I^j_w = 1$ for the x^j with least regret. Constraint (26) ensures R^j_w has its intended meaning; here a is any upper bound on max regret. We also exploit the fact that regret at any w is maximized by the adversary selecting the optimal item x^*_w (x^*_w is precomputed for each w).

Constraint (25) need not be expressed for each (of the continuously many) $w \in W$, but just for the vertices of the feasible space of the MIP's decision variables. Notice, however, that in contrast to single-item regret optimization, where optimality occurs at some vertex $Vert(W)$ of the utility space W, the vertices of the feasible space of the MIP are a superset of $Vert(W)$. More precisely, setwise max regret of a set S can occur not only at points in $Vert(W)$, but at any w that gives the same utility values to several items of the set S, as depicted in Figure 5. Hence, setwise regret of S is maximized at some vertex $\mathcal{V}_S := Vert(W_{S\triangleright x_1}) \cup \ldots \cup Vert(W_{S\triangleright x_k})$, assuming an S-cover of W. It therefore suffices to post constraints only for these vertices. However, this MIP still requires (potentially) exponentially many constraints, one for each element of $\mathcal{V} := \bigcup_{S\subset X:|S|=k} \mathcal{V}_S$.

We can make computation much more effective by applying constraint generation, observing that very few of these constraints will be active. Our procedure works as follows: we solve a relaxed version of the MIP above—the *master problem*—using only the constraints corresponding to a small subset $Gen \subset \mathcal{V}$ of the constraints in the MIP above, and store its optimal value, i.e., the value of objective M. We then test whether any unexpressed constraints are violated at the current solution. This involves computing the true SMR of the recommendation set generated by the master problem. If the SMR of the set is greater than M, we know that a constraint has been violated. Computation of SMR produces the element $w \in \mathcal{V}$ and optimal item x^*_w that corresponds to the maximally violated constraint in the current master solution. We add this maximally violated constraint to *Gen*, tightening the MIP relaxation, and repeat; if no violated constraint exists, we are assured that the current solution minimizes SMR. Note that the adding a new constraint requires the introduction of new variables to the master problem. Every time we add a new w to *Gen*, k new binary variables I and R are necessary.

To solve the setwise max regret subproblem, we can encode the problem as a MIP, adapting the MIP for max regret proposed by Boutilier et al. [21]; given a set S, it computes SMR(S, W) as well as *SMR-AC*(S, W). However, due to Proposition 5, the computation of SMR(S, W) (and the adversarial choice *SMR-AC*(S, W) maximizing setwise regret) can be accomplished by solving k (standard) max regret optimization, and selecting the maximum value. Given a set S, it computes SMR(S, W) as well as *SMR-AC*(S, W). Note that this procedure is also used to construct recommendation/choice sets in the SCAS heuristic below.

Database Problems. Database problems, where options are enumerated in a product database, do not lend themselves to a direct constraint optimization when forming (regret-based) set recommendations since no explicit variable constraints are provided for a MIP, CSP or related formulation. Instead, we adopt a computational model that repeatedly determines the pairwise regret between a candidate recommendation set and an adversarial option in order to identify the option with minimax regret. This can be seen as a game between the recommender and an adversary. The computation of single recommendations is greatly facilitated in practice by formulating the optimization as a minimax search and using standard pruning techniques. However, computing optimal recommendation sets has complexity $O(n^k)$ (for a DB of size n), thus becoming increasingly impractical as the size of the desired set increases. The local search method *query iteration*, SCAS, and CSS (as described below) are best suited to this case.

3.5.2. Heuristics

We now describe other set recommendation and query strategies that exploit minimax regret. Unlike the methods above, these strategies are heuristic and not guaranteed to produce optimal sets. However, they are computationally less demanding than algorithms that guarantee computation of the SMR-optimal option. We use the following notion: define an *adversarial choice* for W and x to be a witness item that maximizes the regret of x:

$$Adv(x, W) \in \arg\max_{x' \in X} \mathrm{MR}(x, x', W).$$

One simple strategy is the *current solution strategy (CSS)* for pairwise comparisons [21, 22]. CSS asks a user to compare two items: the minimax optimal item x^*_W and its adversarial counterpart $Adv(x^*_W, W)$. A pairwise comparison can be viewed as a choice set of size two (thus CSS is restricted to generating sets of size two). User selection of one of these items directly constrains the utility parameters that impact minimax regret.

We now generalize CSS to sets of any size. The *setwise chain of adversaries strategy (SCAS)* selects k options by repeatedly maximizing setwise max regret given the current set. It can be viewed as a greedy, incremental approximation of the (setwise) minimax optimal set. As with setwise max regret, it explicitly maximizes diversity from the perspective of utility space. Formally, SCAS determines a set $\{x^1, \ldots, x^k\}$ where:

$$\begin{cases} x^1 = & x^*_W \\ x^i = & SMR\text{-}AC(\{x^1, .., x^{i-1}\}, W) \;\; 2 \le i \le k \end{cases}$$

Recall that *SMR-AC*(S, W) is an adversarial option w.r.t. setwise max regret. SCAS can be seen as a generalization of the current solution strategy CSS to sets of any size. Computing *SMR-AC*(S, W), which determines a single item at a time, is typically much faster than the joint optimization required by full computation of setwise minimax regret.

Finally, we define the *query iteration (QI) strategy*, an alternative heuristic strategy. The MMR-transformation T introduced above gives rise to a natural heuristic search strategy for construction of good recommendation sets. Given an initial set $S = S_0$, we repeatedly apply T until a fixed point (w.r.t. setwise max regret) is found:

- **Repeat** $S := T(S)$
- **Until** SMR($T(S), W$) = SMR(S, W)

In our experiments (see Section 5.2) we initialize query iteration with the set S given by SCAS, as empirically this seems to produce the most promising sets. In the case where no query can guarantee regret reduction for each response, i.e., WR(S^*_W) = MMR(W), T degenerates to a singleton with only the minimax regret option. In such cases we allow query iteration to return S_0 as determined by SCAS. Note that the query iteration algorithm can be terminated early to allow for anytime query set generation.

Example 1 (continuing from p. 19) *We now consider the applications of the different strategies for generating recommendation sets in the running example. The optimal recommendation set of size two is* $\{x_1, x_4\}$*; SCAS/CSS retrieves*

$\{x_4, x_5\}$ *since SMR-AC*$(\{x_5\}; W) = Adv(x_5; W) = x_4$ *(*x_4 *is the adversarial choice of the minimax optimal item* x_5*); query iteration converges to* $\{x_1, x_4\}$ *(since this set is returned by the operator T applied to any pair of distinct items).*

With $k = 3$*, the optimal set is* $\{x_1, x_3, x_4\}$*. SCAS considers first the minimax regret optimal item* x_5*, it then adds SMR-AC*$(x_5; W) = x_4$ *and finally SMR-AC*$(\{x_4, x_5\}; W) = x_3$*, thus retrieving the set* $\{x_3, x_4, x_5\}$*. Query iteration returns either* $\{x_1, x_2, x_4\}$*,* $\{x_1, x_3, x_4\}$ *or the pair* $\{x_1, x_4\}$ *depending on the starting set. With* $k = 4$*, the optimal recommendation set is* $\{x_1, x_2, x_3, x_4\}$*; SCAS retrieves the set* $\{x_1, x_3, x_4, x_5\}$ *since SMR-AC*$(\{x_3, x_4, x_5\}; W) = x_1$*.*

4. Bayesian Recommendation Sets and Elicitation using Choice Queries

We consider the connection between optimal recommendation sets and optimal choice queries in the context of Bayesian recommendation and utility elicitation, analogous to our approach in the regret-based setting. This section is organized as follows. We begin in Section 4.1 with a discussion of computing optimal *recommendation sets* from the Bayesian point of view. We then formalize the problem of determining the myopically optimal choice query with respect to EVOI in Section 4.2. We outline various noisy response models in Section 4.3, a phenomenon more readily handled in the Bayesian than in the regret-based approach. In Section 4.4, we derive important properties that we use to prove our main theoretical results in Section 4.5. We illustrate key concepts with detailed examples in Section 4.6 and finally present optimization algorithms motivated and supported by our theoretical results in Section 4.7.

4.1. Bayesian Recommendation Sets

We consider first the problem of computing optimal recommendation sets given the system's uncertainty about the user's true utility function w. Given belief state θ, if a single recommendation is to be made, then we should recommend the option $x^*(\theta)$ that maximizes expected utility $\mathrm{EU}(x, \theta)$ (in case of ties, we recommend one of the options with maximum EU). However, there is often value in suggesting a "shortlist" or set containing multiple items and allowing the user to select her most preferred option. Intuitively, such a set should offer options that are *diverse* in the following sense: recommended options should be highly preferred relative to a wide range of "likely" user utility functions [100, 23, 131]. This stands in contrast to some recommender systems that define diversity relative to product attributes [103, 2, 79], with no direct reference to the system's beliefs about user utility. It is easy to see that "top k" systems, those that present the k options with highest expected utility, do not generally result in good recommendation sets [23, 100].

In broad terms, we assume that the utility of a recommendation set S is the utility of its most preferred item. However, it is unrealistic to assume that users will select their most preferred item with complete accuracy [88, 85]. So we assume a *response model* R dictating the probability of specific choices given the user's true (but unknown) utility function w. For any choice set S with $x_i \in S$, let $S \rightsquigarrow x_i$ denote the event of the user selecting x_i. The response model R dictates, for any S, the probability $P_R(S \rightsquigarrow x_i|w)$ of any selection given utility function w. When belief about a user's utility is uncertain, we define the probability of selecting x_i from set S,

$$P_R(S \rightsquigarrow x_i; \theta) = \int_W P_R(S \rightsquigarrow x_i|w)P(w; \theta)dw, \tag{32}$$

by marginalizing $P_R(S \rightsquigarrow x_i|w)$. Specific response models will be thoroughly discussed in Section 4.3. Given a specific selection $S \rightsquigarrow x$, the response model allows us to update the belief state using Bayes' rule:

Definition 15 *The* posterior belief state $\theta|S \rightsquigarrow x$ *w.r.t. prior* θ *and observed choice* $S \rightsquigarrow x$ *is given by*

$$P(w; \theta|S \rightsquigarrow x) = \frac{P_R(S \rightsquigarrow x|w)P(w; \theta)}{\int_{w \in W} P_R(S \rightsquigarrow x|w)P(w; \theta)dw}. \tag{33}$$

We can now define the expected utility of recommendation sets with respect to a response model R:

Definition 16 *The* expected utility of selection (EUS) *of recommendation set S given θ and R is:*

$$\mathrm{EUS}_R(S;\theta) = \sum_{x \in S} P_R(S \rightsquigarrow x;\theta)\,\mathrm{EU}(x;\theta|S \rightsquigarrow x). \tag{34}$$

where $P_R(S \rightsquigarrow x;\theta) = \int_w P_R(S \rightsquigarrow x|w)P(w;\theta)dw$ is the probability of the user selecting item x given belief state θ (cf. Equation 10).

We can restate the definition of $\mathrm{EUS}_R(S;\theta)$ in a way that will be convenient in the sequel:

Proposition 9

$$\mathrm{EUS}_R(S;\theta) = \int_W \Big[\sum_{x\in S} P_R(S \rightsquigarrow x|w)\; u(x;w)\Big]\, P(w;\theta)dw. \tag{35}$$

Proof.

$$\begin{aligned}
\mathrm{EUS}_R(S;\theta) &= \sum_{x\in S} P_R(S \rightsquigarrow x;\theta) \int_W u(x;\ w)\, P(w;\theta|S \rightsquigarrow x)dw \\
&= \sum_{x\in S} P_R(S \rightsquigarrow x;\theta) \int_W u(x;\ w)\, \frac{P_R(S \rightsquigarrow x|w;\ \theta)}{P_R(S \rightsquigarrow x;\theta)} P(w;\theta)dw \\
&= \sum_{x\in S} \int_W u(x;\ w)\, P_R(S \rightsquigarrow x|w)P(w;\theta)dw \\
&= \int_W \sum_{x\in S} \Big[P_R(S \rightsquigarrow x|w)\, u(x;\ w)\Big]P(w;\theta)dw,
\end{aligned}$$

where we first expand the expression of expected utility in the conditioned space, then apply Bayes' rule, then substitute the response model, and finally move the summation inside the integration. Note that $P(w;\theta|S \rightsquigarrow x) = P(w|S \rightsquigarrow x;\theta)$, i.e., the probability of w in the updated belief is the same as probability of w conditioned on choice $S \rightsquigarrow x$ in the original belief, and that $P_R(S \rightsquigarrow x|w;\theta) = P_R(S \rightsquigarrow x|w)$ in the response model gives the likelihood of a user selection given her utility (the user's choice behaviour is completely determined by w). □

As further motivation for our EUS formulation, note that user selection can be seen as a random realization of a categorical distribution, where utilities attached to the individual items are distributed according to $P(w;\theta)$. Equation 35 can then be seen as expressing the expected utility of the selected choice with respect to uncertainty over both w and the item selected by the user: $\mathbb{E}_{w\sim P(w;\theta)}\mathbb{E}_{x\sim P_R(S\rightsquigarrow x;w)}u(x;w)$.

Our goal is to identify the set of size at most k that maximizes expected utility of selection (EUS). We define an *optimal recommendation set* (w.r.t. R) to be any set $S^*_R(\theta)$ of size k maximizing EUS given the current belief θ:

$$\mathrm{EUS}^*_R(\theta) = \max_{S\subseteq X;\,|S|\le k} \mathrm{EUS}_R(S;\theta), \tag{36}$$

$$S^*_R(\theta) \in \arg\max_{S\subseteq X;\,|S|\le k} \mathrm{EUS}_R(S;\theta). \tag{37}$$

We now derive at some basic properties of EUS. For any response model R, the EUS of a singleton $\{x\}$ coincides with the expected utility $\mathrm{EUS}(x;\theta)$ of its only element:

Observation 10 $\mathrm{EUS}_R(\{x\};\theta) = \mathrm{EU}(x;\theta)$ *for any belief θ and $x \in X$.*

Proof. Immediate from Proposition 9, since $P_R(\{x\} \rightsquigarrow x|w) = 1$ for any w (the only one available choice has to be selected). □

Observation 11 *The expected utility of selection of a set S is bounded by the sum of the expected utility of the items in S*

$$\mathrm{EUS}_R(S;\theta) \leq \sum_{x\in S} \mathrm{EU}(x;\theta).$$

Proof. Consider Proposition 9 and the fact that, for any w, $\sum_{x\in S} P_R(S \rightsquigarrow x|w)\, u(x;w) \leq \sum_{x\in S} u(x;w)$ since $P_R(S \rightsquigarrow x|w) \leq 1$. □

Equality $\mathrm{EUS}_R(S;\theta) = \sum_{x\in S} \mathrm{EU}(x;\theta)$ is attained in Observation 11 when the response model is noiseless (see below) and utilities are "orthogonal" in the sense that, for any w with $P(w;\theta) > 0$, there is only one $x \in S$ such that $u(x;w) > 0$.

4.2. Bayesian Elicitation with Choice Queries

The user's response to a choice query tells us something about her preferences; but this depends on the *user response model*. In a *noiseless model*, the user correctly identifies her most preferred item in the set: the choice of $x_i \in S$ refines the set of feasible utility functions W by imposing $k-1$ linear constraints of the form (as in regret-based elicitation, see Section 2.4.2):

$$u(x_i;w) \geq u(x_j;w),\ j \neq i. \tag{38}$$

The new belief state is obtained by restricting θ to have non-zero density only on $W_{S\triangleright x_i}$ and renormalizing.

More generally, a *noisy* response model accounts for the fact that a user may select an option that does not maximize her utility. When treating S as a query set (as opposed to a recommendation set) we are not interested in its expected utility, but rather in its *expected value of information (EVOI)*, or the (expected) degree to which a response will increase the quality of the system's *subsequent* recommendation.

We first define the *expected posterior utility* of a query set (refining Definition 9 specifically for choice queries):

Definition 17 *Given belief state θ, the* expected posterior utility *of query set S under R is*

$$\mathrm{EPU}_R(S;\theta) = \sum_{x\in S} P_R(S \rightsquigarrow x;\theta)\ \mathrm{EU}^*(\theta|S \rightsquigarrow x). \tag{39}$$

The expected value of information $\mathrm{EVOI}_R(S;\theta)$ of a query set S, under response model R, given belief θ is then the difference between EPU_R and the optimal value EU^*, measuring the expected improvement in decision quality given S.

Definition 18 *Given belief state θ, the* expected value of information *of query set S under R is*

$$\mathrm{EVOI}_R(S;\theta) = \mathrm{EPU}_R(S;\theta) - \mathrm{EU}^*(\theta).$$

An optimal query (of fixed size k) is any $S \subset X$ with maximal *EVOI*. If we are looking for the most informative query (with highest EVOI), $\mathrm{EU}^*(\theta)$ can be viewed as a constant; a query set S with maximal expected value of information $\mathrm{EVOI}(S;\theta)$ thus must maximize expected posterior utility $\mathrm{EPU}(S;\theta)$:

Observation 12 *The most informative query set of size at most k is that with highest posterior utility:*

$$\arg\max_{S\subseteq X;\,|S|\leq k} \mathrm{EVOI}_R(S;\theta) = \arg\max_{S\subseteq X;\,|S|=k} \mathrm{EPU}_R(S;\theta). \tag{40}$$

4.3. Probabilistic Response Models

The model of user behavior is largely dictated by the response model R. In the ideal setting, a user would always select the item with greatest utility w.r.t. her true utility function w. This *noiseless model* is assumed in [100] for example. However, this is unrealistic in general. Noisy response models admit user "mistakes," or choice randomness due to exogenous factors, and the choice of optimal sets should reflect this possibility (just as belief update does, see Definition 17). Reasonable requirements on response models include:

1. *Preference bias*: a more preferred item in the slate given w is selected with higher probability than that of a less preferred outcome; and

2. *Luce's choice axiom* [88]: this is a form of *independence of irrelevant alternatives (IIA)* that requires that the relative probability (if not 0 or 1) of selecting any two items x and y from S is not affected by the addition or deletion of other items from the set.[4]

We consider four different response models, each dictating a different form of distribution $P(S \rightsquigarrow x|w)$.

In the *noiseless response model*, R_{NL}, we have that $S \rightsquigarrow x$ implies that x has maximum utility among the items in S. We assume that ties are broken in some consistent, arbitrary (possibly probabilistic) way. Formally, if $x \notin \arg\max_{x \in S} u(x; w)$ then $P_{NL}(S \rightsquigarrow x|w) = 0$.

Observation 13 *Under the noiseless response model* EUS *can be written as*

$$\text{EUS}_{\text{NL}}(S; \theta) = \int_W \max_{x \in S} u(x; w) \;\; P(w; \theta) dw. \tag{41}$$

Proof. Using Proposition 9, this follows from the fact that, for noiseless responses, $P_{NL}(S \rightsquigarrow x|w) = 0$ for $x \notin \arg\max_{x \in S} u(x; w)$ □

EUS_{NL} is identical to the *expected max* (EMAX) criterion considered by Price and Messinger [100].

The *constant noise model* R_C assumes a multinomial distribution over choices or responses where each option x, apart from a most preferred option x^*_w relative to w, is selected with (small) constant probability

$$P_C(S \rightsquigarrow x|w) = \beta \tag{42}$$

with β independent of w. We assume $\beta < \frac{1}{k}$, so a most preferred option is selected with probability

$$P_C(S \rightsquigarrow x^*_w|w) = \alpha = 1 - (k-1)\beta > \beta. \tag{43}$$

Ties are broken randomly: in case of $q > 1$ most preferred options, each is selected with probability $\frac{1-(k-q)\beta}{q}$.

This generalizes the model used by Gajos and Weld [50], Boutilier [18] for comparison queries to query sets of any size. If $x^*_w(S)$ is the optimal element in S given w, and $u^*_w(S)$ is its utility, then EUS under this model can be expressed in a convenient form.

Observation 14 *Under the constant noise model,* EUS *can be written as*

$$\text{EUS}_C(S; \theta) = \int_W \left[\alpha u^*_w(S) + \sum_{y \in S - \{x^*_w(S)\}} \beta u(x; w) \right] P(w; \theta) dw. \tag{44}$$

[4] IIA is one of the more controversial axioms in the theory of choice [127, 85], but a deeper discussion of its pros and cons is beyond the scope of this article.

The *logistic* response model R_L is commonly used in choice modelling, and is variously known as the Luce-Sheppard [87], Bradley-Terry, *mixed multinomial logit* or *conditional logit* model [90, 127]; this model is often used in simulations [59, 123, 124]. Selection probabilities are given by

$$P_L(S \rightsquigarrow x|w) = \frac{e^{\gamma u(x;w)}}{\sum_{y \in S} e^{\gamma u(y;w)}}, \tag{45}$$

where γ is a temperature parameter. For comparison queries (i.e., $|S| = 2$), $P_L(S \rightsquigarrow \cdot|w)$ is the logistic function of the difference in utility between the two options (hence the name we give to this model):

$$P_L(\{x, y\} \rightsquigarrow x|w) = \frac{e^{\gamma[u(x;w)-u(y;w)]}}{1 + e^{\gamma[u(x;w)-u(y;w)]}} = \frac{1}{1 + e^{-\gamma[u(x;w)-u(y;w)]}}. \tag{46}$$

Under this model, a comparison between two items whose difference in utility is $\frac{1}{\gamma}$ corresponds to a probability of a "mistake" of just over 0.26, while a difference $\frac{5}{\gamma}$ corresponds to a mistake probability of less than 0.01. When $\gamma \to \infty$, the logistic model becomes a noiseless response model with ties broken in a (uniform) random way.

Finally we consider, for expository purposes, a degenerate model where responses are given by a uniform discrete distribution, thus selection probabilities do not depend on the underlying utility function. The EUS under this *uniform response model R_U* is simply the average expected utility of the elements of the set. The expected utility of selection given this response model is:

Observation 15

$$\text{EUS}_U(S;\theta) = \frac{1}{k}\sum_{x \in S} EU(x;\theta)$$

where $k = |S|$.

Of course, such a response model is patently unrealistic; but we use it to support the following observation: it can be shown that the optimal set under EUS_U is the set composed of the k items with highest expected utility. In other words *top-k* retrieval (sorting options according to their expected utility), often employed in the information retrieval literature, will in general give sub-optimal recommendation sets, as it implicitly assumes a uniform selection process.

4.4. Properties of EUS *under Different Response Models*

We now consider properties of EUS under these various response models. All three models, R_{NL}, R_C and R_L, satisfy preference bias, but only R_{NL} and R_L satisfy Luce's choice axiom.

Intuitively, the noiseless response model induces greater EUS than any other noise model:

Proposition 16 $\text{EUS}_{\text{NL}}(S;\theta) \geq \text{EUS}_R(S;\theta)$*, for any* $R \in \{\text{NL}, C, L, U\}$.

Proof. Using Observation 13, we obtain

$$\text{EUS}_{\text{NL}}(S;\theta) = \int_W \max_{x \in S} u(x;w)\ \ P(w;\theta)dw \geq \int_W \Big[\sum_{x \in S} P_R(S \rightsquigarrow x|w)\ u(x;w)\Big] P_R(w;\theta)dw = \text{EUS}_R(S;\theta), \tag{47}$$

This holds since, for each $w \in W$, we have $\max_{x \in S} u(x;w) \geq \sum_{x \in S} P_R(S \rightsquigarrow x|w)\ u(x;w)$, which follows from the fact that the maximum utility over all items in S is at least that of than any convex combination of the utility values of the items in S. □

EUS is also monotone under R_{NL}: the addition of options to a recommendation set S cannot decrease its expected utility $\text{EUS}_{\text{NL}}(S;\theta)$:

Proposition 17 $\text{EUS}_{\text{NL}}(S;\theta) \leq \text{EUS}_{\text{NL}}(Q;\theta)$ *with* $S \subseteq Q$.

Proof. Using Observation 13, and the fact $\max_{x \in S} u(x; w) \leq \max_{x \in Q} u(x; w)$, for every $w \in W$, we obtain:

$$\mathrm{EUS}_{\mathrm{NL}}(S;\theta) = \int_W \max_{x \in S} u(x; w) \;\; P(w;\theta)dw \leq \int_W \max_{x \in Q} u(x; w) \;\; P(w;\theta)dw = \mathrm{EUS}_{\mathrm{NL}}(Q;\theta). \tag{48}$$

□

We say that option x_j is *setwise dominated* in S relative to belief θ if for all $w \in W$ such that $P(w;\theta) \geq 0$ there is $x_i \in S$ such that $u(x_i; w) > u(x_j; w)$.

It is easy to see that adding a setwise dominated option x to S does not change S's expected utility under R_{NL}:

Observation 18 $\mathrm{EUS}_{\mathrm{NL}}(S \cup \{x\};\theta) = \mathrm{EUS}_{\mathrm{NL}}(S;\theta)$ *for any x setwise dominated by S.*

This stands in contrast to noisy response models, where adding dominated options might actually *decrease* expected utility.

Importantly, EUS is *submodular* for both the noiseless and the constant response models R_C. We recall the definition of submodularity: a set function f is submodular iff

$$f(A \cup B) + f(A \cap B) \leq f(A) + f(B).$$

An important property is the following: f is submodular iff the "gain" $\rho_x(S) = f(S \cup \{x\}) - f(S)$ of adding an element x to the any S is monotone non-increasing in S for any x:

$$\rho_x(S) \geq \rho_x(Q) \text{ iff } S \subseteq Q. \tag{49}$$

Theorem 19 *For $R \in \{\mathrm{NL}, C\}$, EUS_R is a submodular function of the set S. That is, given two recommendation sets $S \subseteq Q$, item $x \notin S$, $S' = S \cup \{x\}$, and $Q' = Q \cup \{x\}$, we have:*

$$\mathrm{EUS}_R(S';\theta) - \mathrm{EUS}_R(S;\theta) \geq \mathrm{EUS}_R(Q';\theta) - \mathrm{EUS}_R(Q;\theta). \tag{50}$$

Proof. We consider the gain in EUS that sets S and Q incur with the addition of some item $\{a\}$. Let $\rho(S;\theta) = \mathrm{EUS}(\{a\} \cup S;\theta) - \mathrm{EUS}(S;\theta)$. We write $\rho(S;\theta) = \int_W p(w)\rho(S;w)dw$ in order to make explicit the contribution $\rho(S;w)$ of each w in the gain $\rho(S;\theta)$ (cf. Proposition 9):

$$\rho(S;w) = P_R(S' \rightsquigarrow a|w)u(a;w) + \sum_{x \in S} \Big[P_R(S' \rightsquigarrow x|w) - P_R(S \rightsquigarrow x|w)\Big]u(x;w), \tag{51}$$

where $S' = S \cup \{a\}$ and $Q' = Q \cup \{a\}$. We will show that, for each w, $\rho(S;w) \geq \rho(Q;w)$. From this, it follows $\rho(S;\theta) \geq \rho(Q;\theta)$ (characteristics of diminishing returns), whatever the belief θ, as the summation or integration of submodular functions is also submodular.

Noiseless response model (NL): Recall that $x^*_w(S)$ denotes the (true) optimal item in set S given utility weights w. We consider different cases with respect to the optimality of item $\{a\}$ in the set $S' = S \cup \{a\}$ and $Q' = Q \cup \{a\}$.

First, if w is such that a is optimal in S' and optimal in Q', $\rho(S', w) = u(a;w) - u^*_w(S) \geq u(a;w) - u^*_w(Q) = \rho(Q', w)$, since $u^*_w(Q) \geq u^*_w(S)$. Second, suppose w is such that a is optimal in S', but not optimal in Q'; then, $\rho(S', w) = u(a;w) - u^*_w(S) \geq 0 = \rho(Q', w)$. Third, suppose w is such that a is neither optimal in S', nor in Q': $\rho(Q', w) = \rho(S', w) = 0$ (the gain is zero for both sets). Finally, the following case is impossible: w such that a optimal in Q', but not optimal in S' (impossible because $S' \subseteq Q'$). Thus EUS with noiseless responses is submodular.

Constant noise response model (C): We substitute $P_R(S \rightsquigarrow x|w)$ in the expression of the gain according to the constant response model. Most terms in the summation in Equation 51 evaluate to β and cancel out. Note that the likelihood α of a correct answer is a function of the set size (when there are more items, it is easier for the user to make a mistake). Let $\alpha_S = \alpha(|S|) = 1 - (|S| - 1)\beta$ (similarly for S', Q and Q'), it holds $\alpha_{S'} = \alpha_S - \beta$ and $\alpha_{Q'} = \alpha_Q - \beta$.

Again, we consider different cases with respect to the optimality of item $\{a\}$ in the set $S' = S \cup \{a\}$ and $Q' = Q \cup \{a\}$, and write the expression of the gain in each case.

1. Suppose w is such that a is optimal in S' and optimal in Q':

$$\rho(S', w) = \alpha_{S'}[u(a; w) - u^*_w(S)] \geq 0 \tag{52}$$
$$\rho(Q', w) = \alpha_{Q'}[u(a; w) - u^*_w(Q)] \geq 0 \tag{53}$$

2. Suppose w such that a is optimal in S', but not optimal in Q'. S has a gain, while Q a loss.

$$\rho(S', w) = \alpha_{S'}[u(a; w) - u^*_w(S)] \geq 0 \tag{54}$$
$$\rho(Q', w) = \beta[u(a; w) - u^*_w(Q)] \leq 0 \tag{55}$$

3. Suppose w such that a is neither optimal in S', nor in Q'. The gain will be negative for both sets (thus a loss).

$$\rho(S', w) = \beta[u(a; w) - u^*_w(S)] \leq 0 \tag{56}$$
$$\rho(Q', w) = \beta[u(a; w) - u^*_w(Q)] \leq 0 \tag{57}$$

4. The following case is impossible: w such that a optimal in Q', but not optimal in S' (impossible because $S' \subseteq Q'$).

In all cases $\rho(S; w) \geq \rho(Q; w)$, i.e., the gain is greater for S than for the larger set Q (in the third case, S suffers a smaller loss): by componentwise comparison, and the fact that $u^*_w(Q) \geq u^*_w(S)$ and $\alpha_{S'} > \alpha_{Q'}$ and that, for the model to be meaningful we have $\alpha > \beta$. □

The proof shows that EUS has the required property of diminishing returns. Submodularity serves as the basis for a greedy optimization algorithm (see Section 4.7). EUS under the commonly used logistic response model R_L is not submodular, but can be related to EUS under the noiseless model—as we discuss in Section 4.5.3 —allowing us to exploit submodularity of the noiseless model when optimizing w.r.t. R_L.

4.5. Theoretical results: the Connection between EUS *and* EPU

We develop the connection between optimal recommendation sets (with respect to EUS) and optimal choice queries (with respect to EPU/EVOI). As discussed above, we are often interested in sets that can serve as both good recommendations and good queries; and since computing or optimizing EPU/EVOI can be computationally difficult, good methods for EUS-optimization can serve to generate good queries as well if we have a tight relationship between the two.

We first provide a characterization of the optimal query set under noiseless responses in Section 4.5.1. We then analyze the impact of several noisy response models in Sections 4.5.2 and 4.5.3. Our results can be summarized as follows:

- For the noiseless response model, the optimal recommendation set of size k is also an optimal query set of size k.
- For the constant noise model, the optimal recommendation set of size at most k is also the optimal query set of size at most k.
- For the logistic model, logistic noise has a bounded effect on the value of a set, hence, optimal recommendation sets are near-optimal queries—we derive a closed-form expression for such a bound.

First of all, we remark that, for a given response model R and belief θ, EUS is a lower bound for EPU; in other words, the value of a set as a query is always at least its value as a recommendation set; this follows directly from the definition of EUS (Eq.34) and EPU (Eq. 17):

Proposition 20 $\mathrm{EPU}_R(S; \theta) \geq \mathrm{EUS}_R(S; \theta)$.

Proof. Since $\mathrm{EU}(x;\, \theta | S \rightsquigarrow x) \leq \mathrm{EU}^*(\theta | S \rightsquigarrow x)$,

$$\mathrm{EUS}_R(S;\, \theta) = \sum_{x \in S} P_R(S \rightsquigarrow x; \theta)\, \mathrm{EU}(x; \theta | S \rightsquigarrow x) \leq \sum_{x \in S} P_R(S \rightsquigarrow x; \theta)\, \mathrm{EU}^*(\theta | S \rightsquigarrow x) = \mathrm{EPU}_R(S;\, \theta).$$

□

In the following, we make use of a transformation $T_{\theta,R}$ that modifies a set S in such a way that EUS usually increases (and in the case of R_{NL} and R_C cannot decrease). This transformation is used in two ways: (i) to prove the optimality (near-optimality in the case of R_L) of EUS-optimal recommendation sets when used as query sets; and (ii) directly as a computationally viable heuristic strategy for generating query sets.

Definition 19 *Let $S = \{x_1, \cdots, x_k\}$ be a set of options. Define:*

$$T_{\theta,R}(S) = \{x^*(\theta | S \rightsquigarrow_R x_1), \cdots, x^*(\theta | S \rightsquigarrow_R x_k)\}$$

where $x^*(\theta | S \rightsquigarrow_R x_i)$ is an optimal option (in expectation) when θ is conditioned on $S \rightsquigarrow x_i$ w.r.t. R: $x^*(\theta | S \rightsquigarrow_R x_i) \in \arg\max_{x \in X} \mathrm{EU}(x; \theta | S \rightsquigarrow_R x)$.

Intuitively, T (we drop the subscript when θ, R are clear from context) refines a recommendation set S of size k by producing k updated beliefs for each possible user choice, and replacing each option in S with the optimal option (with respect to expected utility) under the corresponding update. Note that $T(S)$ may be of lesser cardinality than S, since different beliefs may induce the same optimal option. In general, there might be more than one optimal option in an updated belief; for ease of notation, we assume $T(S)$ breaks ties in some (potentially) non-deterministic way. We use $\tau(S)$ whenever we want to explicitly refer to the set of all possible sets $T(S)$ that could be constructed in this way.

Below, we make use of the idea of a "fixed point" of T, defined as any set S such that $S \in \tau_\theta(S)$. If S is a fixed point of T, then

$$\mathrm{EUS}(S; \theta) = \mathrm{EUS}(T_{\theta,R}(S); \theta). \tag{58}$$

It is important to notice (even if technically straightforward to prove) the following:

Proposition 21 *Consider a set S as a query set and assume that S is a fixed point for T, i.e., $S \in \tau_\theta(S)$. If a user selects x from S, then x is an optimal (single) recommendation w.r.t. the posterior belief $\theta | S \rightsquigarrow x$, i.e.,*

$$x \in \arg\max_{y \in X} \mathrm{EU}(y; \theta | S \rightsquigarrow x).$$

The proof is immediate from the definition of the T operator and from the fact that S is a fixed point.

Note that T generally produces different sets under different response models. Indeed, one could use T to construct a set using one response model, and measure EUS or EPU of the resulting set under a different response model. Some of our theoretical results use this type of "cross-evaluation."

We now use T to provide a formulation of EPU in an integral form (similar to Proposition 9 for EUS).

Proposition 22 *Let $S = \{x_1, \cdots, x_k\}$ be a set of options, and let $x'_i \in \arg\max_{x \in X} \mathrm{EU}(x; \theta | S \rightsquigarrow x_i)$; that is the items resulting from the application of the operator $T_{\theta,R}$. EPU_R of set S can be written as:*

$$\mathrm{EPU}_R(S; \theta) = \int_W \sum_{i=1}^{k} [P_R(S \rightsquigarrow x_i | w) u(x'_i; w)] P(w; \theta) dw \tag{59}$$

Note that, we could have $x'_i = x'_j$ for $i \neq j$ if the same item is optimal in two different posteriors (i.e. if the cardinality of $T(S)$ is less than k) and the utility of the same item would be counted twice in the inner summation of Equation 59.

Proof. We know that $x'_i \in \arg\max_{x \in X} \mathrm{EU}(x; \theta | S \rightsquigarrow x_i)$.

$$\mathrm{EPU}_R(S;\theta) = \sum_{i=1}^{k} P_R(S \rightsquigarrow x_i;\theta)\ \mathrm{EU}^*(\theta | S \rightsquigarrow x_i) = \sum_{i=1}^{k} P_R(S \rightsquigarrow x_i;\theta)\ \mathrm{EU}(x'_i | S \rightsquigarrow x_i) \tag{60}$$

$$= \sum_{i=1}^{k} P_R(S \rightsquigarrow x_i;\theta) \int_W u(x'_i;\ w)\, P(w;\theta | S \rightsquigarrow x_i) P(w;\theta) dw \tag{61}$$

$$= \sum_{i=1}^{k} P_R(S \rightsquigarrow x_i;\theta) \int_W u(x'_i;\ w)\, \frac{P_R(S \rightsquigarrow x_i | w;\ \theta)}{P_R(S \rightsquigarrow x_i;\theta)} P(w;\theta) dw \tag{62}$$

$$= \sum_{i=1}^{k} \int_W u(x'_i;\ w)\, P_R(S \rightsquigarrow x_i | w;\ \theta) P(w;\theta) dw \tag{63}$$

$$= \int_W \sum_{i=1}^{k} [u(x'_i;\ w)\, P_R(S \rightsquigarrow x_i | w;\ \theta)]\, P(w;\theta) dw. \tag{64}$$

□

We now turn to our main results in the Bayesian setting. We address noiseless responses next, and defer discussion of noisy responses to Section 4.5.2 (for the constant noise model) and Section 4.5.3 (for the logistic noise model).

4.5.1. The Connection between EUS *and* EPU *under Noiseless Responses*

We now show that optimal recommendation sets are optimal (i.e., EPU/EVOI-maximizing) query sets under noiseless responses. We make use of the T operator in the following proposition, which we dub the *Bayesian Query Iteration lemma* (compare the analogous Lemma 7 in the regret-based framework), as it plays a central role in proving our main results.

Lemma 23 (*Bayesian Query Iteration under Noiseless Responses*) $\mathrm{EUS}_{NL}(T_{\theta,\mathrm{NL}}(S);\theta) \geq \mathrm{EPU}_{\mathrm{NL}}(S;\theta)$.

Proof. Let $S = \{x_1, \cdots, x_k\}$ be a set of options, and let $x'_i \in \arg\max_{x \in X} \mathrm{EU}(x;\theta | S \rightsquigarrow x_i)$ for $i \leq k$. Let $T_{\theta,NL}(S) = \{x'_1, \cdots, x'_k\}$ be the set resulting from the application of the T operator with respect to the noiseless response model (we drop θ and NL in the subscript of T below for clarity). We formulate EUS and EPU (under noiseless responses) according to Observation 13 and Proposition 22, respectively, and derive

$$\mathrm{EUS}_{\mathrm{NL}}(T(S);\theta) = \int_W \max_{i=1,\dots,k} u(x'_i; w)\ \ P(w;\theta) dw \geq \int_W \sum_{i=1}^{k} [P_{\mathrm{NL}}(S \rightsquigarrow x_i | w) u(x'_i; w)] P(w;\theta) dw = \mathrm{EPU}_{\mathrm{NL}}(S;\theta). \tag{65}$$

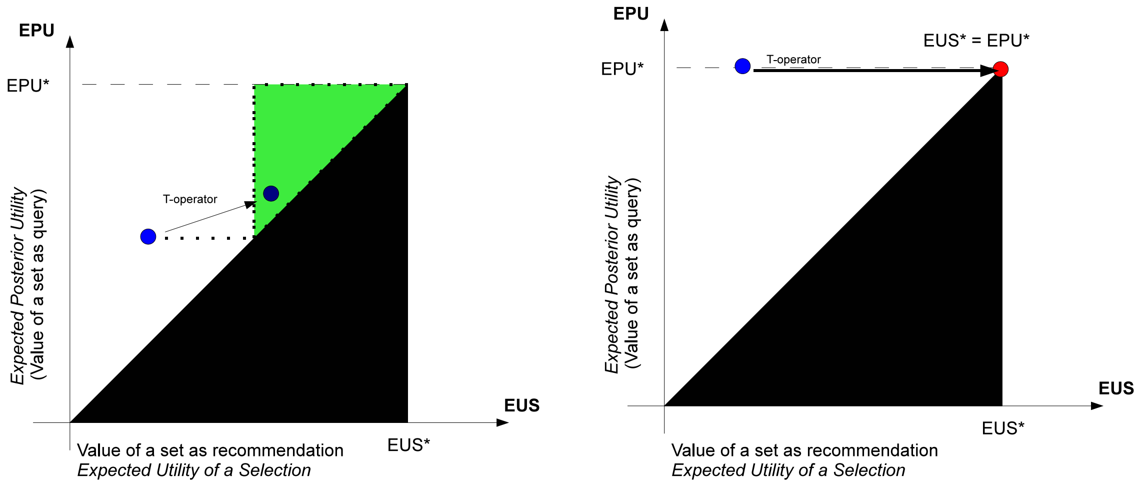
Equation 65 holds because, for each w, $\max_{i=1,\dots,k} u(x'_i; w) \geq \sum_{i=1}^{k} [P_{\mathrm{NL}}(S \rightsquigarrow x_i | w) u(x'_i; w)]$ since the maximum among $u(x'_1; w), \dots, u(x'_k; w)$ is at least the value of any convex combination of utility values $u(x'_1; w), \dots, u(x'_k; w)$.

Note that, even if $T_{\theta,NL}(S)$ has fewer than k items, $\mathrm{EUS}_{\mathrm{NL}}(T(S);\theta)$ is equivalent to $\int_W \max_{i=1,\dots,k} u(x'_i; w)\ \ P(w;\theta) dw$.
□

From Lemma 23 and Proposition 20 we have the following corollary:

Corollary 24 *Under the noiseless response model R_{NL} we have:*

- $\mathrm{EUS}_{\mathrm{NL}}(T_{\theta,NL}(S);\ \theta) \geq \mathrm{EUS}_{\mathrm{NL}}(S;\ \theta)$*; and*
- $\mathrm{EPU}_{\mathrm{NL}}(T_{\theta,NL}(S);\ \theta) \geq \mathrm{EPU}_{\mathrm{NL}}(S;\ \theta)$.


EPU
EPU*
T-operator
Expected Posterior Utility
(Value of a set as query)
EUS
EUS*
Value of a set as recommendation
Expected Utility of a Selection
EUS* = EPU*

(a) Points represents sets in the EUS/EPU space (b) An optimal recommendation set is also an optimal query

Figure 6: The intuition behind the proof: using the Query Iteration Lemma, we show that the non-existence of a set that is both optimal as a recommendation and as a query leads to contradiction. There are no sets in the shaded area as EUS is a lower bound for EPU

This means that, given the current belief θ, the result of applying T to a given set S is a new set that is at least as good as S as a recommendation set (with respect to EUS), and at least as good as S when considered as a query set (with respect to EPU). The Query Iteration Lemma 23 states something even stronger: *the value of the set obtained by applying T as a set of recommendations (w.r.t.* EUS*) is at least the value of the original set S as query (w.r.t.* EPU*).* This lies at the heart of the reasoning behind the main theorem for noiseless responses, asserting that a set S^* that is optimal with respect to EUS is also an EPU-optimal query set.

Theorem 25 *Assume the noiseless response model* NL *and let S^* be an optimal recommendation set of size k. Then*

- *S^* is an optimal query set of size k:* $\text{EPU}_{\text{NL}}(S^*;\theta) \geq \text{EPU}_{\text{NL}}(S;\theta),\ \forall S \in 2^X, |S| = k$*; and*
- *The value of the optimal set as recommendation coincides with its value as a query:* $\text{EUS}_{\text{NL}}(S^*;\theta) = \text{EPU}_{\text{NL}}(S^*;\theta)$ *and therefore* $\text{EUS}^*_{\text{NL}}(\theta) = \text{EPU}^*_{\text{NL}}(\theta)$.

Proof. Note that all EUS/EPU computations in this proof are with respect to NL, so we drop the subscript. Suppose S^* is not an optimal query set, i.e., there is some S s.t. $\text{EPU}(S;\theta) > \text{EPU}(S^*;\theta)$. Applying T to S gives a new query set $T(S)$, which by Lemma 23 shows: $\text{EUS}(T(S);\theta) \geq \text{EPU}(S;\theta) > \text{EPU}(S^*;\theta) \geq \text{EUS}(S^*;\theta)$. This contradicts the EUS-optimality of S^*.

To prove the second part, note that as S^* is an optimal recommendation set, $\text{EUS}(S^*;\theta) \geq \text{EUS}(T(S^*);\theta)$. By Lemma 23, $\text{EUS}(T(S^*);\theta) \geq \text{EPU}(S^*;\theta)$, and therefore $\text{EUS}(S^*;\theta) \geq \text{EPU}(S^*;\theta)$. But since it holds $\text{EPU}(S^*;\theta) \geq \text{EUS}(S^*;\theta)$ by Proposition 20, we conclude that $\text{EUS}^*(\theta) = \text{EUS}(S^*;\theta) = \text{EPU}(S^*;\theta) = \text{EPU}^*(\theta)$. □

Consider the optimal recommendation set. While one might suppose that a weaker recommendation set could yield better value as a query, our results show that this is not possible: the operator T provides an operational mechanism for producing a set that is optimal as both a recommendation set and as a choice query. Somewhat surprisingly, the decision-theoretic "diversity" of the optimal recommendation set is (myopically) maximally informative for elicitation.

The intuition behind the proof is illustrated graphically in Figure 6. In Figure 6(a) each candidate set is represented by a point in the EUS/EPU space. The T operator is such that the output set $T(S)$ is better with respect to both criteria (EUS and EPU) than the original set (in Figure 6(a) this is the green area). Assume there is an optimal query set that it is not an optimal recommendation set. By applying the operator T, the resulting set must be optimal with respect to both EUS and EPU (see Figure 6(b)).

Another consequence of Lemma 23 is that posing a query S involving an infeasible option is pointless: there is always a set with only elements in X whose EPU/EVOI at least as great. This is proven by observing the Query Iteration Lemma still holds if T is redefined to allow sets containing infeasible options. This is an observation of important practical interest since asking about preferences concerning hypothetical items is often criticized by behavioral decision scientists and practitioners. The fact that the best query (with respect to our myopic criterion) is free to use only real items is therefore notable.

Corollary 26 $\mathrm{EUS}_{\mathrm{NL}}(S;\theta) = \mathrm{EUS}_{\mathrm{NL}}(T(S);\theta)$ *is a necessary condition for the optimality of set S under belief θ with respect to both* EUS *and* EPU.

The fact that applying the operator T cannot decrease the quality of a set (Lemma 23), and the fact that a fixed point is required for optimality (Corollary 26) justify it as an operational way to improve a set. This will be exploited by the query iteration strategies described below in Section 4.7, where we develop algorithms for set optimization (to be used either as recommendations or as queries).

The condition $S \in \tau_\theta(S)$ is also a necessary condition for the optimality of set S. The fact that $S \in \tau_\theta(S)$ is not a sufficient condition can be seen by considering the singleton set $S_d = \{x^*_\theta\}$ containing the item with highest expected utility $x^*_\theta = \arg\max_{x\in X} \mathrm{EU}(x;\theta)$. We have $T(S_d) = S_d$, but obviously S_d will be in general suboptimal as a recommendation/query set.

Another consequence of our main result, Theorem 25, is the following. Consider the situation where the system has an opportunity to ask exactly one query and, after observing the user's response, must produce a final recommendation. This situation can be represented by an influence diagram: the root is a decision node for choosing a query; the query response is modelled by a chance node; and a final decision node (one for each of the possible responses) models the provision of a recommendation from X. The solution of this influence diagram is a policy that can be represented by a decision tree of width 1. At the root node, the optimal decision is to ask the query with maximal EUS (and therefore maximal EPU). According to Corollary 26, the optimal query set with respect to the initial belief is a fixed point of T. Proposition 21 guarantees that, at any fixed point, the best recommendation at the posterior belief is exactly the item selected by the user (as the response to the only query the system could pose). This situation is studied in Appendix A and used to represent the solution of the famous Monty Hall problem in what we believe to be an novel formulation of this problem.

4.5.2. The Connection between EUS *and* EPU *under the Constant Noise Model*

We now explore the connection between optimal query sets and optimal recommendation sets given a response model with constant noise. Under R_C, we obtain that the optimal recommendation set is an optimal query set, as in the noiseless case.

Critically, we derive a version of the Query Iteration Lemma (cf. Lemma 23) for the constant noise model:

Lemma 27 (*Bayesian Query Iteration under Constant Noise*) $\mathrm{EUS}_C(T_{\theta,\mathrm{NL}}(S);\theta) \geq \mathrm{EPU}_C(S;\theta)$.

Proof. As usual, let $S = \{x_1, \ldots, x_k\}$ and $T(S) = \{x'_1, \ldots, x'_k\}$. Let $x^*_w(S) = \arg\max_{x\in S} u(x;w)$ and $u^*_w(S) = \max_{x\in S} u(x;w)$. Using Proposition 14 and Proposition 22, we have:

$$\mathrm{EUS}_C(T(S);\theta) = \int_W \left[\alpha u^*_w(T(S)) + \beta \sum_{x'_j \in T(S), x'_j \neq x^*_w(T(S))} u(x'_j)\right] P(w;\theta)dw. \tag{66}$$

$$\mathrm{EPU}_C(S;\theta) = \int_W \sum_{i=1}^{k} [P(S \rightsquigarrow x_i|w) u(x'_i;w)] P(w;\theta)dw = \int_W \left[\alpha u(x'_{f(S;w)}) + \beta \sum_{j\neq f(S,w)} u(x'_j)\right] P(w;\theta)dw. \tag{67}$$

Here $f(S;w) = \arg\max_{i\in\{1,\ldots,k\}} u(x_i;w)$ is the index of item with highest utility in S. We now show that, for all $w \in W$, $\alpha u^*_w(T(S)) + \beta \sum_{x'_j\in T(S), x'_j\neq x^*_w(T(S))} u(x'_j) \geq \alpha u(x'_{f(S;w)}) + \beta \sum u(x'_j)$.

- If w is such that $S \rhd x_i$ and $T(S) \rhd x'_i$, then the left hand side and the right hand side have the same value.

- If w is such that $S \rhd x_i$ and $T(S) \rhd x'_j$, with $i \neq j$, then we have $\alpha u^*_w(T(S)) + \beta u(x'_{f(S,w)}; w) \geq \alpha u(x'_{f(S,w)}; w) + \beta u^*_w(T(S))$, since $u^*_w(T(S)) > u(x'_{f(S,w)}; w)$ and $\alpha > \beta$.

Since for each $w \in W$ the contribution to $\mathrm{EPU}(S;\theta)$ is no greater than that of $\mathrm{EUS}(T(S);\theta)$, the lemma follows. □

As in the noiseless response model, we have the following property (cf. Corollary 24).

Corollary 28 *Under the constant noise model R_C, we have:*

- $\mathrm{EUS}_C(T_{\theta,C}(S);\theta) \geq \mathrm{EUS}_C(S;\theta)$*, and*
- $\mathrm{EPU}_C(T_{\theta,C}(S);\theta) \geq \mathrm{EPU}_C(S;\theta)$.

This corollary allows us to use T as a fast method for locally improving a set (in the same way as in the noiseless response model), an insight we exploit algorithmically in Section 4.7.

We now establish the connection between optimal recommendations sets and optimal query sets. Under the constant noise model, the optimal recommendation set of size at most k is also the optimal query set of size at most k. This weaker form of the theorem is due to the fact that T might produce a set of smaller cardinality (this is not a concern in the noiseless model, as EUS is monotone).

Theorem 29 *Assume a constant noise response model and let S^* be an optimal recommendation set of size at most k, i.e., $S^* \in \arg\max_{S \subseteq X;\, |S| \leq k} \mathrm{EUS}_C(\theta)$. Then:*

- *S^* is optimal query set of size at most k:* $\mathrm{EPU}_C(S^*;\theta) \geq \mathrm{EPU}_C(S;\theta)$, $\forall S \subset X;\ |S| \leq k$.
- *The value of S^* as recommendation coincides with its value as a query:* $\mathrm{EUS}_C(S^*;\theta) = \mathrm{EPU}_C(S^*;\theta)$ *and therefore* $\mathrm{EUS}^*_C(\theta) = \mathrm{EPU}^*_C(\theta)$.

The proof is essentially the same as the proof of Theorem 25 but making use of Lemma 27 (rather than Lemma 23).

4.5.3. The Connection between EUS *and* EPU *under the Logistic Noise Model*

We now explore the connection between selecting an optimal query set and an optimal recommendation set given a response model with logistic noise. Under R_L, we first derive a bound on the effect of logistic noise on the quality of set recommendations, and we then exploit this bound to prove that EUS-optimal recommendation sets are near-optimal choice queries.

The value of a recommendation set measured by the EUS criterion decreases when choices are "noisy" as the selected item is not necessarily that with greatest utility among those in the set. For example, under the logistic noise model we have $\mathrm{EUS}_L(S;\theta) \leq \mathrm{EUS}_{NL}(S;\theta)$, and generally this inequality is strict. We are interested in the degree to which noisy responses under R_L impact the value of a recommendation set relative to noiseless responses under R_{NL}; i.e., what is the loss in expected utility of selection for a given a recommendation set due to logistic noise:

Definition 20 *The loss in expected utility of recommendation set S given belief θ induced by logistic response noise is:*

$$\Delta(S;\theta) = \mathrm{EUS}_{\mathrm{NL}}(S;\theta) - \mathrm{EUS}_L(S;\theta). \tag{68}$$

Notice that two sources of uncertainty arise in $\Delta(S;\theta)$: the uncertainty in the user's response dictated by R_L; and the uncertainty due to the system's limited belief θ about the user's utility. This loss can be express as:

$$\Delta(S;\theta) = \mathbb{E}_{w\sim P(w;\theta)}\Big[u^*_w(S) - \mathbb{E}_{x\sim P_L(S\rightsquigarrow\cdot;w)}u(x;w)\Big], \tag{69}$$

where $u^*_w(S) = \max_{x\in S} u(x;w)$ is the utility of the most preferred item in S if the "true" user utility is w. It is useful to study the effect of the two sources of uncertainties separately. We first show that the uncertainty due to the user's response *assuming a known utility function* has a relatively limited effect on this loss. We then assess the impact of utility uncertainty and show that this also bounds $\Delta(S;\theta)$. With such reasoning, we can bound the difference between $\mathrm{EUS}_{\mathrm{NL}}$ and EUS_L in a way that depends only on the set cardinality k and the (logistic) temperature parameter γ.

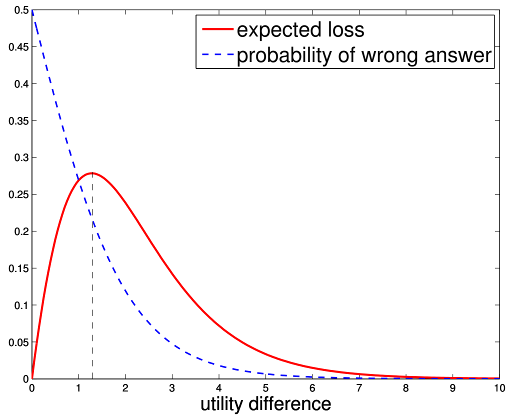
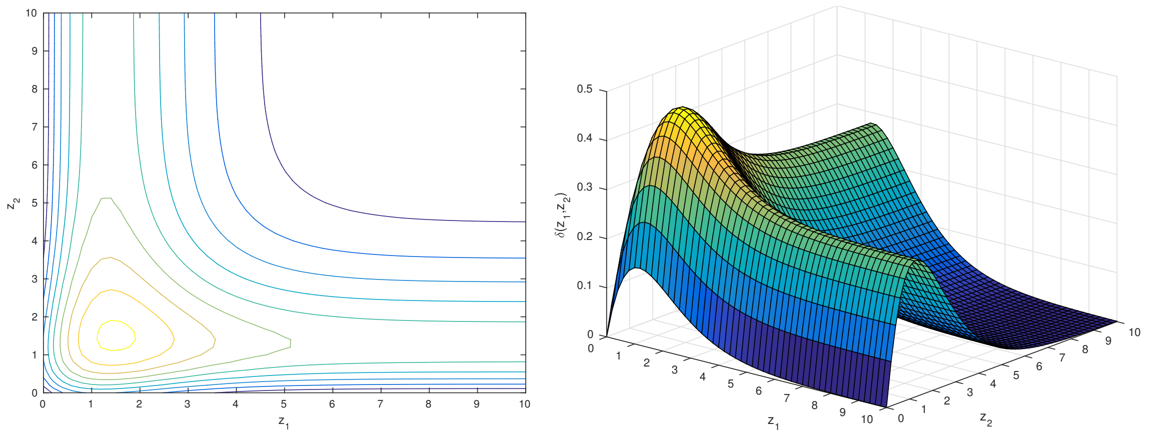


Figure 7: Expected loss δ as a function of the utility difference z, for pairs of items ($k = 2$), with $\gamma = 1$. The loss is maximum for $\hat{z} = 1.279$, with $\hat{\Delta}^2 = 0.279$ and associated with a probability of around 0.22 of making an incorrect answer.



(a) Contour plot of $\delta(z_1, z_2)$

(b) Surface plot of $\delta(z_1, z_2)$

Figure 8: Countour and surface plots of the expected loss for the case $k = 3$ as a function of z_1 and z_2 (the utility difference between the two worst items and the best one).

Expected loss due to user choice. Let $x^*_w(S) = \arg\max_{x\in S} u(x)$ be the utility-maximizing item in S assuming user utility w. The expected loss due to logistic noise (for a fixed w) is the probability of the user selecting a suboptimal item times the utility difference between the selected item and $x^*_w(S)$:

$$\delta(S) = u^*_w(S) - \sum_{x\in S} P_L(S \rightsquigarrow x; w)u(x; w) = \sum_{x\in S} P_L(S \rightsquigarrow x; w)\Big[u^*_w(S) - u(x; w)\Big] \tag{70}$$

Note that loss due to user choice can be seen as computing $\Delta(S\,; \theta)$ (see Equation 68) in the degenerate case where the belief state θ has its mass concentrated on a single w.

We now analyze Equation 70. We first consider the case $k = 2$ with two items x_1, x_2 for ease of exposition. Let $v_1 = u(x_1; w) > v_2 = u(x_2; w)$ be the utility of the items and $z = v_1 - v_2$ their difference. When $z > 0$, loss $\delta(\{x_1 x_2\})$ due to user choice is z times the probability of choosing the less preferred item x_2 under R_L:

$$(v_1 - v_2)P_L(S \rightsquigarrow x_2; w) = (v_1 - v_2)\frac{e^{\gamma v_2}}{e^{\gamma v_1} + e^{\gamma v_2}} = \frac{z}{1 + e^{\gamma z}} \tag{71}$$

We write the expected loss, for the case $k = 2$, as $\delta(z)$ to make explicit that the expected loss depends only on z. Inspecting Equation 71, we observe that the expected loss under R_L can be bounded. As z increases, the probability

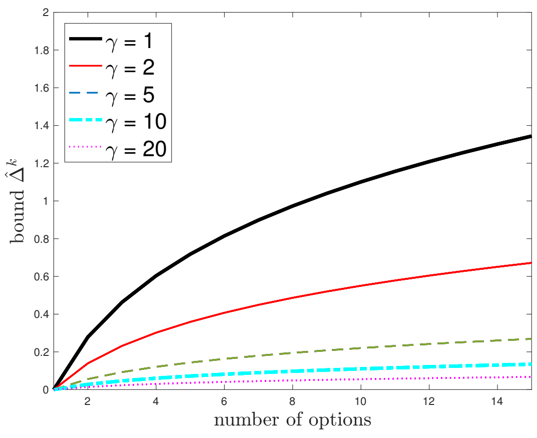


Figure 9: Plot of the worst-case bound $\hat{\Delta}^k$ for the logistic noise model for different values of γ and different item-set sizes k.

of selecting the less preferred item decreases, but the impact (i.e., the utility difference z) of this "incorrect" choice increases. The loss can thus be bounded by $\max_{z \in \mathbb{R}^+} \delta(z)$ at corresponding value $\hat{z}$. Figure 7 plots (in red) the expected loss as function of z for $\gamma = 1$—the maximum loss $\hat{\Delta}^2 \approx 0.279$ (for $k = 2$) is attained at $\hat{z} \approx 1.279$. In the same figure, we also plot (in blue) the curve $\frac{1}{1+e^{\gamma z}}$, representing the probability of selecting the less preferred item when $\gamma = 1$; such an incorrect response has probability ≈ 0.22 for the loss-maximizing value $\hat{z}$. We express the bound in analytical form, for any γ, making use of the *Lambert W function* [40], also known as *product-log*:[5]

$$\hat{z} = \frac{1}{\gamma} \cdot \left[1 + \mathcal{L}\left(\frac{1}{e}\right)\right] \approx 1.279\frac{1}{\gamma}, \tag{72}$$

$$\hat{\Delta}^2 = \delta(\hat{z}) = \frac{1}{\gamma} \cdot \mathcal{L}\left(\frac{1}{e}\right) \approx 0.279\frac{1}{\gamma}. \tag{73}$$

Here $\mathcal{L}$ denotes the Lambert W function.

Similar reasoning applies when the choice set size is $k > 2$. The loss δ can be formulated as a function of the utility differences between the options in the set. Indeed, the contribution of each item in S to the expected loss is the utility loss w.r.t. $x^*_w(S)$ times its selection probability. Assuming, without loss of generality, that the kth item is that with greatest utility ($v_k \geq v_i$ for all $i \in \{1, \ldots, k\}$), the expected loss due to user choice is:

$$\delta(v_1, \ldots, v_k) = \sum_{i=1}^{k} (v_k - v_i) \frac{e^{v_i}}{\sum_{j=1}^{k} e^{v_j}} = \sum_{i=1}^{k-1} (v_k - v_i) \frac{1}{1 + \sum_{j \neq i} e^{v_j - v_i}}. \tag{74}$$

The expected loss δ can be expressed as a function of the terms $v_j - v_i$. We set $z_i = v_k - v_i$, which must be at least 0 (given the assumption that v_k is the highest value); in this way, we are left with a problem with $k-1$ free variables, representing the utility loss of each item with respect to the kth item. We write $\delta(z_1, \ldots, z_{k-1})$ to make it explicit that the expected loss depends only on z_i:

$$\delta(z_1, \ldots, z_{k-1}) = \sum_{i \in K^-} z_i \frac{1}{1 + e^{z_i} + \sum_{j \neq i} e^{z_i - z_j}} \tag{75}$$

where $K^- = \{1, \ldots, k-1\}$. Figure 8 shows the plot of this function for the case $k = 3$ with $\gamma = 1$; the maximum in this case is ≈ 0.463 attained when $z_1 = z_2 \approx 1.463$. In general, *the maximum loss is always attained with a set S in*

[5]Lambert W is defined as the principal value of the inverse of xe^x, where e^x is the exponential function. It allows one to write the solution of an equation $xe^x = y$ as $x = \mathcal{L}(y)$. While the Lambert W function is generally multi-valued, here we consider a real-valued positive domain and a real-valued codomain, in which case it is single-valued.

which all items except the optimal item $x^*_w(S)$ have the same utility. In other words, the utility difference between the $x^*_w(S)$ and all other items is the same: the loss-maximizing $(\hat{z}_1, \ldots, \hat{z}_{k-1})$ is such that $\hat{z}_1 = \ldots = \hat{z}_{k-1}$, which allows us to derive a bound $\hat{\Delta}^k$ on this loss, as shown in the following Theorem:

Theorem 30 *Assume a set selection problem with known user utility, in which the user with utility w—inducing item utilities $v_1, \ldots, v_k$—selects an item from slate S (of size k). The expected loss induced by user choice under response model R_L is at most:*

$$\hat{\Delta}^k = \frac{1}{\gamma} \cdot \mathcal{L}(\frac{k-1}{e}), \tag{76}$$

where $\mathcal{L}$ is the Lambert W function [40]. Moreover, the maximum loss is attained by utilities $v_1, \ldots, v_k$:

$$v_1 = \ldots = v_{k-1} = \zeta \tag{77}$$

$$v_k \qquad = \zeta + \frac{1}{\gamma} \cdot \left[1 + \mathcal{L}\left(\frac{k-1}{e}\right)\right] \tag{78}$$

with $\zeta \in \mathbb{R}$, or any permutation of these values (since the loss does not depend on the ordering of the values $v_1, \ldots, v_k$).

Proof. See Appendix B. □

Figure 9 shows the value of $\hat{\Delta}^k$ for different values of γ and different numbers of options.

Example 2 *Consider the following three situations each involving a binary choice with a known utility function under the logistic response model.*

1. *The choice between a good item (utility 1) and a bad item (utility 0).*
2. *The choice is between two good items (utilities 1 and 0.99, resp.).*
3. *The choice between an above-average item (utility 0.66) and a below-average item (utility 0.33).*

We consider temperatures $\gamma = 3$ (high noise) and $\gamma = 5$ (mild noise). Key to user choice (and induced loss Δ) under R_L are the utility differences (respectively, 1, 0.01 and 0.33 in the three choice situations), rather than the magnitude of the individual values.

In the first choice setting, the loss due to user choice of the suboptimal item is high (loss of 1), but the difference in utility is such that this event is unlikely: with $\gamma = 3$ the probability of selecting the best item is 0.95, so expected loss due to user choice is only $\Delta = 0.0474$; with $\gamma = 5$, the probability of selecting the most-preferred item increases to 0.99 and expected loss drops to $\Delta = 0.0067$.

In the second case, user choice is more "difficult," as both choices are good. The probability of picking the suboptimal item is high (approximately 0.49), but the loss in this case is only 0.01, so expected loss is $\Delta = 0.0049$; with $\gamma = 5$, the probability of correct response increases only marginally and Δ remains close to 0.0049.

Finally, in the third case, the probability of the optimal choice with $\gamma = 3$ (resp., $\gamma = 5$) is 0.73 (resp., 0.84) and $\Delta = 0.0894$ (resp., $\Delta = 0.0532$). This choice setting has the highest expected loss among the three.

We compare these loss values with the bound of Theorem 30: we have $\hat{\Delta}^2 = 0.0928$ (for $\gamma = 3$) and $\hat{\Delta}^2 = 0.0557$ ($\gamma = 5$); these values are attained for utility differences between the two items of $\hat{z} = 0.4262$ and $\hat{z} = 0.2557$, respectively. Notice that the third choice example with a utility difference of 0.33 induces losses due to user choice that are quite close to these bounds.

Expected loss due to utility uncertainty. We now show how the bound $\hat{\Delta}^k$ derived in Theorem 30 can be used to bound the expected loss in the more general case where the system's belief state θ about the user's utility function w is uncertain. This requires integration over possible utility functions, or equivalently, over possible z values (i.e., utility differences). Since probabilities must sum to one, this loss is maximized when all probability mass is concentrated on utility functions corresponding to $\hat{z}$ computed above. This allows one to derive a problem-independent upper bound $\hat{\Delta}^k$ on the expected loss $\Delta(S\,;\theta)$—for any S, θ—that depends only on k.

Note that the belief distribution $P(w;\,\theta)$ induces a distribution $P(z_1,\ldots,z_{k-1};\,\theta)$ over the utility differences for all items in the set. This gives us the following relationship between the expected loss Δ and the loss δ due to user choice:

$$\Delta(S;\,\theta) = \int_{z_1} \ldots \int_{z_{k-1}} \delta(z_1,\ldots,z_{k-1})\, P(z_1,\ldots,z_{k-1};\,\theta)\, dz_1 \ldots dz_{k-1}. \tag{79}$$

From this, we derive the following bound:

Theorem 31 $\mathrm{EUS}_L(S;\theta) \geq \mathrm{EUS}_{\mathrm{NL}}(S;\theta) - \hat{\Delta}^k$, *where*

$$\hat{\Delta}^k = \frac{1}{\gamma}\cdot\mathcal{L}(\frac{k-1}{e}),$$

where $\mathcal{L}(\cdot)$ is the Lambert W function.

Proof. Consider $k = 2$. We show that the expected loss of any set can be at most the expected loss associated with two hypothetical items x_1 and x_2 having the same utility difference $u(x_1, w) - u(x_2, w) = \hat{z}$ for any $w \in W$.

$$\Delta(S;\theta) = \mathrm{EUS}_{NL}(S;\theta) - \mathrm{EUS}_L(S;\theta) = \int_{z=-\infty}^{z=+\infty} \delta(z)P(z;\theta)dz = \int_{-\infty}^{+\infty} |z|\cdot\frac{1}{1+e^{\gamma|z|}}P(z;\theta)dz \leq \hat{z}\frac{1}{1+e^{\gamma\hat{z}}} = \hat{\Delta}^2$$

The reasoning is analogous for $k > 2$. □

Note that the loss-maximizing set $S_{\max}$ typically contains infeasible outcomes; so in practice, the actual realizable loss may be much lower. The particular form of the bound allows us to bound the difference between $\mathrm{EUS}_{\mathrm{NL}}$ and EUS_L at some value $\hat{\Delta}_k$ that depends only on the set cardinality k and the temperature parameter γ.

By plugging Equation 76 into the formula for logistic noise with binary choices (Equation 46), we can be quantify the loss on a scale that is independent of the temperature parameter γ.

Theorem 32 *The bound $\hat{\Delta}^k$ corresponds to a utility difference such that in a hypothetical choice situation between a pair of alternatives, under R_L with temperature γ, the user reports her preference incorrectly with probability*

$$\frac{1}{1+e^{\mathcal{L}(\frac{k-1}{e})}}.$$

Note that this value does not depend on γ.

Proof. Assume an hypothetical choice between two pairs with utility difference $z = \hat{\Delta}^k = \frac{1}{\gamma}\cdot\mathcal{L}(\frac{k-1}{e})$. The probability of making an incorrect choice under R_L is $\frac{1}{1+e^{\gamma z}} = \frac{1}{1+e^{\mathcal{L}(\frac{k-1}{e})}}$. □

This last theorem can be appreciated by considering the following thought experiment: imagine a recommender faced with a choice situation where the difference in utility between the two items (x^+ and x^-, with x^+ more preferred to x^-) is exactly $\hat{\Delta}^k$ and the user is answering according to the logistic noise model (with the same γ as in the original decision context). One might expect the bound to correspond to a utility value large enough to discriminate x^+ from x^-. However, Theorem 32 states that, when $k = 2$, the value $\hat{\Delta}$ corresponds to a utility difference so slight that the user identifies the best item only with probability 0.57. In other words, *the maximum loss is so small that the user is unable able to identify the preferred item 43% of the time when asked to compare the two items x^+, x^-* (whose difference in utility, by construction, is exactly the bound). For larger values of k, the probability of incorrect selection decreases, but it remains surprisingly low: $k = 10$ induces incorrect selection with probability of roughly 0.25; and to drive the odds of incorrect selection under 10% requires $k = 55$.

Now that we have established a bound on the impact of logistic noise, we can establish a connection between EUS and EPU, i.e., between the value of a set viewed as recommendation set and as a query set. We proceed in a similar fashion as in the other noise models, using the operator T that improves the quality of a set S by considering the EU-optimal items in the posterior belief associated with S (viewed as a query), and establishing a query iteration lemma. In R_L however, our results employ the bound $\hat{\Delta}$ to show that optimal recommendation sets are *near-optimal* choice queries.

Under R_L, our transformation T_L does not, in general, improve the value $\mathrm{EUS}_L(S)$ of a recommendation set S. However the set $T_L(S)$ is such that its value $\mathrm{EUS}_{\mathrm{NL}}$, *assuming selection under the noiseless model*, is greater than the expected posterior utility $\mathrm{EPU}_L(S)$ under R_L:

Lemma 33 (*Bayesian Query Iteration under Logistic Noise*) $\mathrm{EUS}_{\mathrm{NL}}(T_L(S);\theta) \geq \mathrm{EPU}_L(S;\theta)$.

Proof. The proof is very similar to that of Lemma 23. Let $x'_i = x^*(\theta|S \rightsquigarrow x_i)$ under R_L; that is, $x'_i \in \arg\max_{x \in X} \mathrm{EU}(x;\theta|S \rightsquigarrow x_i)$. $T_L(S)$ is the set composed of $x'_1, \ldots, x'_k$ with duplicates removed. Note that even if the cardinality of $T_L(S)$ is less than k, we have $EUS_{\mathrm{NL}}(T_L(S);\theta) = \int_W \max_{i=1,\ldots,k} u(x'_i;w) \;\; P(w;\theta)dw$. We derive

$$\mathrm{EUS}_{\mathrm{NL}}(T_L(S);\theta) = \int_W \max_{i=1,\ldots,k} u(x'_i;w) \;\; P(w;\theta)dw \geq \int_W \sum_{i=1}^{k} [P_L(S \rightsquigarrow x_i|w) u(x'_i;w)] P(w;\theta)dw = \mathrm{EPU}_L(S;\theta). \quad (80)$$

Equation 80 holds because, for each w, $\max_{i=1,\ldots,k} u(x'_i;w) \geq \sum_{i=1}^{k} [P_L(S \rightsquigarrow x_i|w) u(x'_i;w)]$; that is, $u^*_w(T_L)$ is at least a great as any convex combination of the utilities $u(x'_j;w)$.

□

We use this fact below to prove that the optimal recommendation set under R_L is a near-optimal query set under R_L (see Theorem 36). However, we require two additional lemmas. First, from Theorem 31 and Lemma 33, we derive:

Lemma 34 $\mathrm{EUS}_L(T_L(S);\theta) \geq \mathrm{EPU}_L(S;\theta) - \hat{\Delta}^k$.

Second, we can relate $\mathrm{EPU}^*_{\mathrm{NL}}(\theta)$ to $\mathrm{EPU}^*_L(\theta)$ as follows. In general, $\mathrm{EPU}_L(S;\theta)$ may be greater than $\mathrm{EPU}_{\mathrm{NL}}(S;\theta)$: there are sets for which a noisy response might be "more informative" than a noiseless one. However, this is not the case for *optimal* query sets. One consequence of Lemma 33 is that the EPU of the optimal query under R_{NL} is at least as great as that of the optimal query under R_L:

Lemma 35 $\mathrm{EPU}^*_{\mathrm{NL}}(\theta) \geq \mathrm{EPU}^*_L(\theta)$.

Proof. Let q^*_L be the optimal query set w.r.t. belief θ under logistic response: $q^*_L = \arg\max_q \mathrm{EPU}_L(q;\theta)$ and $\mathrm{EPU}^*_L(\theta) = \mathrm{EPU}_L(q^*_L;\theta)$. Dropping θ from our notation for clarity, we have:

$$\mathrm{EPU}^*_{\mathrm{NL}} = \mathrm{EUS}^*_{NL} \geq \mathrm{EUS}_{\mathrm{NL}}(T_L(q^*_L)) \geq \mathrm{EPU}_L(q^*_L) = \mathrm{EPU}^*_L. \quad (81)$$

□

We now derive our main result for the logistic response model: the EUS of the optimal recommendation set of size at most k is at most $\hat{\Delta}^k$ less than the EPU of the optimal query set (dropping k and γ from the Δ-notation for clarity). This implies that the optimal recommendation set S^* can be used as query set whose value $\mathrm{EPU}_L(S^*)$ is within $\hat{\Delta}$ of the optimal query value EPU^*_L.

Theorem 36 $\mathrm{EUS}^*_L(\theta) \geq \mathrm{EPU}^*_L(\theta) - \hat{\Delta}^k$.

Proof. Consider the optimal query S^*_L and the set $S' = T_L(S^*_L)$ obtained by applying T_L. From Lemma 33, we have:

$$\mathrm{EUS}_{\mathrm{NL}}(S';\theta) \geq \mathrm{EPU}_L(S^*_L,\theta) = \mathrm{EPU}^*_L(\theta).$$

From Theorem 31, we obtain:

$$\mathrm{EUS}_L(S';\theta) \geq \mathrm{EUS}_{\mathrm{NL}}(S';\theta) - \hat{\Delta}^k;$$

and from Theorem 25, $\mathrm{EUS}^*_{\mathrm{NL}}(\theta) = \mathrm{EPU}^*_{\mathrm{NL}}(\theta)$. Thus

$$\mathrm{EUS}^*_L(\theta) \geq \mathrm{EUS}_L(S';\theta) \geq \mathrm{EUS}_{\mathrm{NL}}(S';\theta) - \hat{\Delta}^k \geq \mathrm{EPU}^*_L(\theta) - \hat{\Delta}^k.$$

□

We summarize the theoretical results for the three response models in Table 3.

Response model	EUS is monotone	EUS is submodular	EUS lower bound for EPU	optimal set is optimal query EUS* = EPU*	T_R improves both EUS and EPU
NL	yes	yes	yes	yes	yes
C	no	yes	yes	yes	yes
L	no	no	yes	no, but optimal EUS set is a near-optimal query (loss at most $\hat{\Delta}$)	no, but decrease is at most $\hat{\Delta}$

Table 3: Summary of theoretical results relating optimal recommendation sets and optimal query sets under the noiseless (NL), constant noise (C) and logistic noise (L) response models.

.

4.6. Illustrations

We now provide several examples to illustrate the results above.

Example 3 *We consider an elicitation problem developed by Price and Messinger [100], where different laptop configurations are evaluated according to two features, battery life and processor speed. Four items x_1,x_2,x_3, and x_4 have the following feature values:*

$$x_1 = (0.05, 1), \quad x_2 = (0.5, 0.6), \quad x_3 = (0.6, 0.5), \quad x_4 = (1, 0.05).$$

Note that x_2 and x_3 are more "moderate" options while x_1 and x_4 are "extreme." Utility is assumed to be linear, so $u(x; w) = w[1]x[1] + w[2]x[2]$ for utility function $w = (w[1], w[2])$.

We consider an extreme situation where only two user types/profiles are possible: *video editors*, who want maximum raw power and do not care about battery life, and have utility weights $w_1 = (0, 1)$; and *executive travellers*, who care only about battery life, with weights $w_2 = (1, 0)$. The prior over the two user types is uniform. Under this prior, we have the following expected utilities:

Alternative	x_1	x_2	x_3	x_4
EU	0.525	0.55	0.55	0.525

The best alternatives are x_2 and x_3, both achieving EU* = 0.55.

Under noiseless response, EUS for all two-item recommendation sets is as follows:

	Recommendation/Query Sets					
	$\{x_1, x_2\}$	$\{x_1, x_3\}$	$\{x_1, x_4\}$	$\{x_2, x_3\}$	$\{x_2, x_4\}$	$\{x_3, x_4\}$
EUS_{NL}	0.75	0.8	1.0	0.6	0.8	0.75

The set $\{x_1, x_4\}$ is the best according to the EUS criterion—indeed, since there are only two realizable utility profiles, the best 2-set must include the optimal item for each user type. Notice that neither x_2 nor x_3, the items achieving EU* for single-item recommendations, belong to the EUS-maximizing 2-set. This illustrates that recommending the top k items w.r.t. (individual) expected utility is generally suboptimal as a recommendation set (a point made in [100]). In this example the set $\{x_2, x_3\}$ has very low EUS = 0.6.

Greedy optimization of the 2-set using EUS—see Section 4.7 below for details—will select x_2 or x_3 as first element, and then select x_4 (resp., x_1) as second element as these provide the maximal increase EUS *conditional on* x_2 (resp., x_3) being the first item, therefore obtaining EUS = 0.8—a greater value than the top-k approach.

When treating 2-sets as queries, since there are only two feasible utility functions, any set of two elements has EPU = 1 (hence, EVOI = 0.45), as any such query will rule out one of the two utility functions and allow immediate

	Recommendation/Query Sets					
	$\{x_1, x_2\}$	$\{x_1, x_3\}$	$\{x_1, x_4\}$	$\{x_2, x_3\}$	$\{x_2, x_4\}$	$\{x_3, x_4\}$
EUS_{NL}	0.683	0.717	**0.841**	0.584	0.716	0.684
EPU_{NL}	**0.841**	**0.841**	**0.841**	**0.841**	0.828	0.828
EVOI_{NL}	**0.29**	**0.29**	**0.29**	**0.29**	0.277	0.277

Table 4: EUS and EPU values of all the possible pairs of items under noiseless responses (Example 3)

identification of the item with highest utility. It is also easy to see that applying T_{NL} to any pair of distinct items will converge to the optimal set $\{x_1, x_4\}$ in one step.

Suppose we introduce a third user type, *balanced users* with utility weights $w_3 = (0.52, 0.48)$, and assume that all three types have the same prior probability. Notice that under w_3 we have $x_3 \succ x_2 \succ x_4 \succ x_1$. With this change in prior distribution, item x_3 is optimal with $\text{EU}^* = 0.551$. Under noiseless responses, EUS and EPU for all two-item recommendation/query sets are shown in Table 4; the optimal 2-set as both recommendation and query set is $\{x_1, x_4\}$ with EVOI = 0.29; several other 2-sets achieve maximal EVOI without being an optimal recommendation set. Notice that EVOI does not improve expected posterior utility to 1—unlike the two-profile case, the answer to a query of size two cannot fully discriminate between w_1, w_2 and w_3. We note also that the application of T_{NL} to any pair returns the optimal set $\{x_1, x_4\}$.

Considering recommendation and query sets of size three, optimal 3-set is clearly $\{x_1, x_3, x_4\}$—the set consisting of the three items that are optimal with respect to w_1, w_2 and w_3. It has EUS and EPU of 0.851 (and EVOI = 0.30). The application of operator T to any triplet, other than $\{x_1, x_2, x_3\}$, immediately yields the optimal set $\{x_1, x_3, x_4\}$. Interestingly, $T_{\text{NL}}(\{x_1, x_2, x_3\}) = \{x_1, x_4\}$ gives a resulting set of lesser cardinality since x_2 is strictly less preferred than x_1 and x_3 for each of w_1, w_2, w_3 (cf. Definition 19).

Usually we are not given an explicit enumeration of a finite set of valid utility functions (as in the example above). More often utility parameters belong to a (bounded) continuous space, as in the next example.

Example 4 *We reconsider the Example 1, used in Section 3.4 to illustrate minimax regret, in the Bayesian context. Item feature values are recalled in the table below, along with expected utilities. We assume utility function $u(x; w) = w \cdot (x[1] - x[2]) + x[2]$, with $w \in [0, 1]$, and a uniform utility prior $w \sim U[0, 1]$.*

	$x_i[1]$	$x_i[2]$	EU
x_1	0.35	0.68	0.515
x_2	0.9	0.2	0.550
x_3	0	0.75	0.375
x_4	1	0	0.500
x_5	0.5	0.3	0.400

The best recommendation under this prior (with no elicited preferences) is x_2 and $\text{EU}^* = 0.55$.

Table 5 shows EUS, EPU and (myopic) EVOI under noiseless responses for all 2-item sets. The optimal recommendation set is $\{x_1, x_4\}$, which is also the optimal query set, with myopic EVOI = 0.124. Note that $\text{EUS}_{NL}(\{x_1, x_4\}) = \text{EPU}(\{x_1, x_4\})$ as guaranteed by Theorem 25.

The result of applying the operator T (assuming noiseless responses) to any of the pairs above is as follows:

Set S	$\{x_1, x_2\}$	$\{x_1, x_3\}$	$\{x_1, x_4\}$	$\{x_1, x_5\}$	$\{x_2, x_3\}$	$\{x_2, x_4\}$	$\{x_2, x_5\}$	$\{x_3, x_4\}$	$\{x_3, x_5\}$	$\{x_4, x_5\}$
$T_{NL}(S)$	$\{x_1, x_4\}$	$\{x_2, x_3\}$	$\{x_1, x_4\}$	$\{x_1, x_4\}$	$\{x_1, x_4\}$	$\{x_1, x_4\}$	$\{x_2, x_3\}$	$\{x_1, x_4\}$	$\{x_1, x_4\}$	$\{x_1, x_4\}$

	Recommendation/Query Sets									
	$\{x_1,x_2\}$	$\{x_1,x_3\}$	$\{x_1,x_4\}$	$\{x_1,x_5\}$	$\{x_2,x_3\}$	$\{x_2,x_4\}$	$\{x_2,x_5\}$	$\{x_3,x_4\}$	$\{x_3,x_5\}$	$\{x_4,x_5\}$
EUS_{NL}	0.662	0.521	**0.674**	0.536	0.654	0.567	0.560	0.661	0.507	0.556
EPU_{NL}	0.672	0.622	**0.674**	0.646	0.662	0.658	0.631	0.669	0.673	0.662
EVOI_{NL}	0.123	0.072	**0.124**	0.096	0.112	0.108	0.081	0.119	0.123	0.112

Table 5: EUS and EPU values of all the possible pairs of items under noiseless responses (Example 4)

	Recommendation/Query Sets									
	$\{x_1,x_2\}$	$\{x_1,x_3\}$	$\{x_1,x_4\}$	$\{x_1,x_5\}$	$\{x_2,x_3\}$	$\{x_2,x_4\}$	$\{x_2,x_5\}$	$\{x_3,x_4\}$	$\{x_3,x_5\}$	$\{x_4,x_5\}$
	R_C with "medium" noise ($\beta = 0.1$)									
EUS_C	0.636	0.506	**0.641**	0.520	0.616	0.558	0.543	0.616	0.483	0.535
EPU_C	0.639	0.599	**0.641**	0.618	0.633	0.628	0.607	0.637	0.640	0.633
EVOI_C	0.089	0.049	**0.091**	0.068	0.083	0.078	0.057	0.087	0.090	0.083
	R_L with "medium" noise ($\gamma = 10$)									
EUS_L	0.647	0.502	**0.662**	0.517	0.643	0.546	0.542	0.651	0.490	0.539
EPU_L	0.653	0.597	**0.662**	0.606	0.652	0.584	0.604	0.662	0.650	0.635
EVOI_L	0.103	0.047	**0.112**	0.056	0.102	0.034	0.054	0.112	0.100	0.085
	R_L with "very high" noise ($\gamma = 1$)									
EUS_L	**0.555**	0.454	0.544	0.467	0.512	0.527	0.486	0.502	0.406	0.466
EPU_L	0.555	0.550	0.560	0.550	0.564	0.550	0.550	**0.570**	0.553	0.550
EVOI_L	0.005	0	0.010	0	0.014	0	0	**0.020**	0.003	0

Table 6: EUS and EPU values of all the possible pairs of items under noisy responses (Example 4).

The set $\{x_1, x_4\}$ is the only fixed point in this example. Query iteration converges to the best set $\{x_1, x_4\}$ in at most 2 steps. For instance, starting from $\{x_1, x_3\}$, the application of T leads to the set $\{x_2, x_3\}$, and from the latter, another application of T gives $\{x_1, x_4\}$.

We now consider noisy responses and show EUS and EPU for all 2-sets in Table 6. For the constant noise model R_C, we assume $\beta = 0.1$, and consider two cases, $\gamma = 10$ ("mildly noisy") and $\gamma = 1$ ("very noisy"), in the logistic noise model R_L.

With constant noise, $\{x_1, x_4\}$ is the optimal recommendation set and is optimal as a set query (with respect to EPU_C) as guaranteed by Theorem 29. Notice, as expected, its values are lower than with the noiseless response model.

Set $\{x_1, x_4\}$ is again optimal in the logistic model with "mild" noise $\gamma = 10$, both as a recommendation set and a query set. The loss due to noise is $\text{EUS}_{\text{NL}}(\{x_1, x_4\}) - \text{EUS}_L(\{x_1, x_4\}) = 0.012 < \hat{\Delta}^k \approx 0.027$, according to Theorem 31. Set $\{x_2, x_4\}$ has the greatest loss ($\Delta(S\,; \theta) = 0.021$) with respect to noiseless responses.

In the logistic model with "high" noise $\gamma = 1$, set $\{x_1, x_2\}$ is optimal with respect to EUS_L while the set $\{x_3, x_4\}$ is the optimal query with respect to EPU_L. If we consider the pair $\{x_1, x_2\}$, its loss $\text{EUS}_{\text{NL}}(\{x_1, x_2\}) - \text{EUS}_L(\{x_1, x_2\}) = 0.107$ is, of course, less than the worst-case bound $\Delta \approx 0.279$ offered by Theorem 31. We also verify Theorem 36: $\text{EPU}^*_L - \text{EUS}^*_L = \text{EPU}_L(\{x_3, x_4\}) - \text{EUS}_L(\{x_1, x_2\}) = 0.015 < 0.279$. Finally note that the best set under noiseless responses R_{NL} is also a good set in this setting: $\{x_1, x_4\}$ as a recommendation set is second-best, worse than the optimal set $\{x_1, x_2\}$ by only 0.011. Interestingly, it is better than the latter as a query—in fact it is within 0.010 of the optimal query set. We note that EVOI_L in the high-noise logistic model is quite small, with some query pairs having zero value of information, due to the rather significant impact of such extreme noise.

4.7. Algorithms

We now provide algorithms for computing recommendations sets with the Bayesian uncertainty model. We first consider the computational complexity of exact optimization of recommendation sets. Price and Messinger [100] analyze the complexity of EMAX—a setwise decision criterion that is formally equivalent to EUS_{NL}—with respect to set size k and show that the optimization of a recommendation set is NP-complete from this point of view. Since optimization of EUS with a noisy response model is a more general problem, the optimization remains intractable with respect to k.

By contrast, in this work, we are interested in computation time with respect to *the number of available choices*, and treat k as a small constant (e.g., single digits).[6] We believe this is a more practically relevant perspective, since most recommender systems have constraints on the number of items that can be "comfortably" be displayed, presented or recommended to a user in single slate— hence k will be typically be small. However, the number of items available from which to choose when making such a recommendation will often be in the dozens, hundreds, or many thousands. As such, tractability relative to the size of the candidate set is of more practical importance.

We develop several algorithmic strategies for the optimization of query/recommendation sets in this section, and analyze their theoretical and computational properties. In what follows, $|X| = n$ is the number of candidate options, k the size of the query/recommendation set, and l is the "cost" of Bayesian inference (e.g., the number of particles in a Monte Carlo sampling procedure).

Exact Methods. The naïve maximization of EPU is more computationally intensive than EUS-optimization, and is generally impractical. Given a set S of k elements, computing $\mathrm{EPU}(S, \theta)$ requires Bayesian update of θ for each possible response, and expected utility optimization for each such posterior, with complexity $O(nkl)$. Query optimization requires that we compute this for $\binom{n}{k}$ possible query sets of size k. Consequently, naïve EPU maximization has a worst-case running time of $O(n^{k+1}kl)$.

Exact EUS optimization by iterating over all possible subsets of size k, while still quite demanding, is "only" $O(n^k kl)$ as it does not require EU-maximization in updated distributions. Theorem 25 allows us to compute optimal query sets using EUS-maximization under R_C and R_{NL}, reducing complexity by a factor of n. Under R_L, Theorem 36 allows us to use EUS-optimization to approximate the optimal query, with a quality guarantee of $\mathrm{EPU}^* - \hat{\Delta}$.

Greedy Optimization. A simple greedy algorithm can be used to generate a recommendation set of size k by iteratively adding the option x' offering the greatest improvement in value:

$$x' \in \arg\max_{x \in X} \mathrm{EUS}_R(S \cup \{x\}; \theta).$$

Since EUS is submodular (Theorem 19) the greedy algorithm determines a near-optimal set with worst-case guarantees in the case of noiseless responses.

Theorem 37 *Under R_{NL} the greedy algorithm determines a set with both EUS and EPU that are within $\eta = 1 - (\frac{k-1}{k})^k$ of the optimal value $EUS^* = EPU^*$.*

Proof. This follows directly from the fact that EUS is monotone and submodular (Theorem 19) and from the seminal results of Nemhauser, Wolsey and Fisher [96]. Theorem. 25 again allows us to use greedy maximization of EUS to determine a *query set* with similar guarantees. □

[6]When larger sets are displayed, they are often organized into categories, topics, "shelves," or the like, which we view as very different than an unstructured set recommendation. Similarly, while scrolling interfaces one can be interpreted as presenting a recommendation set of *dynamic*, and potentially large, size to a user, we also view this as having more structure than a simple recommendation set, since the act of scrolling over (skipping) recommendations early in a dynamic set provides information that can be used to dynamically reset the choice of subsequent items that appear later.

This bound is 75% for comparison queries ($k = 2$) and at worst 63% as $k \to \infty$.

For R_C, EUS_C is submodular but not monotone. One can apply a randomized greedy optimization algorithm (with the same running time as greedy) [28], with a worst-case bound of $\frac{1}{e}$ that holds *in expectation with respect to the internal randomness of the algorithm.*

Theorem 38 *Under R_C the randomized greedy algorithm determines a set with both EUS and EPU that are within $\frac{1}{e}$ of the optimal value $EUS^* = EPU^*$.*

The proof follows by a direct application of the result of Buchbinder *et al.* [28].

Under R_L, EUS_L is no longer submodular. However, noiseless optimization is effective even when evaluated in a noisy setting. We derive a bound on the EUS of the greedy set when evaluated in a setting with logistic response noise. Similarly, we derive a worst-case bound for EPU with respect to greedy EUS-optimization.

Theorem 39 *Greedy maximization of* $\mathrm{EUS}_{\mathrm{NL}}$ *gives near-optimal recommendation/elicitation sets under logistic response noise. Specifically, Let S_g be the set determined by greedy optimization of* $\mathrm{EUS}_{\mathrm{NL}}$. *Then*

$$\frac{\mathrm{EUS}_L(S_g)}{\mathrm{EUS}^*_L} \geq \eta - \frac{\hat{\Delta}^k}{\mathrm{EUS}^*_{NL}} \quad \textit{and} \quad \frac{\mathrm{EPU}_L(S_g)}{\mathrm{EPU}^*_L} \geq \eta - \frac{\hat{\Delta}^k}{\mathrm{EUS}^*_{NL}},$$

where $\eta = 1 - (\frac{k-1}{k})^k$.

Proof. Lemma 33 and Theorem 31 allow us to use $\mathrm{EUS}_{\mathrm{NL}}$, which is submodular, as a proxy. By Theorem 37, $\eta \cdot \mathrm{EUS}^*_{\mathrm{NL}} \leq \mathrm{EUS}_{\mathrm{NL}}(S_g) \leq \mathrm{EUS}^*_{NL}$; we also have $\mathrm{EUS}^*_L \leq \mathrm{EUS}^*_{\mathrm{NL}}$. Applying Theorem 31 to S_g gives: $\mathrm{EUS}_L(S_g) \geq \mathrm{EUS}_{NL}(S_g) - \hat{\Delta}^k$. Thus, we derive

$$\frac{\mathrm{EUS}_L(S_g)}{\mathrm{EUS}^*_L} \geq \frac{\eta \cdot \mathrm{EUS}^*_{NL} - \hat{\Delta}^k}{\mathrm{EUS}^*_L} \geq \frac{\eta \cdot \mathrm{EUS}^*_{NL} - \hat{\Delta}^k}{\mathrm{EUS}^*_{NL}} \geq \eta - \frac{\hat{\Delta}^k}{\mathrm{EUS}^*_{NL}}. \tag{82}$$

Similarly, for EPU, we use the fact that EUS is a lower bound for EPU, Observation 20, Theorem. 31 and Lemma 35 and Theorem. 25:

$$\frac{\mathrm{EPU}_L(S_g)}{\mathrm{EPU}^*_L} \geq \frac{\mathrm{EUS}_L(S_g)}{\mathrm{EPU}^*_L} \geq \frac{\eta \cdot \mathrm{EUS}^*_{NL} - \hat{\Delta}^k}{\mathrm{EPU}^*_{NL}} = \frac{\eta \cdot \mathrm{EUS}^*_{NL} - \hat{\Delta}^k}{\mathrm{EUS}^*_{NL}} \geq \eta - \frac{\hat{\Delta}}{\mathrm{EUS}^*_{NL}}. \tag{83}$$

□

Greedy maximization of S w.r.t. EUS is extremely fast, $O(k^2 ln)$. Most critically, it is linear in the number of options n, requiring just $O(kn)$ evaluations of EUS, each with cost kl.

Lazy greedy EUS *optimization.* In addition of providing strong worst-case guarantees on the quality of the approximation, submodularity allows us to use a speedup known as *lazy evaluation* [93]). Given the current belief θ, we want to produce a set of recommendations with greatest EUS. The greedy algorithm is such that, at each round, it picks the element that maximizes $\mathrm{EUS}(S \cup x; \theta)$ or, equivalently, the greatest marginal benefit:

$$\rho_S(x) = \mathrm{EUS}(S \cup x; \theta) - \mathrm{EUS}(S; \theta). \tag{84}$$

The greedy algorithm can be made more efficient using lazy evaluation. At round $i+1$ of the greedy maximization of EUS, assume we have a set $S_i = \{x_1, .., x_i\}$, and must add an $i+1$st item x_{i+1} that maximizes EUS, $x_{i+1} = \arg\max_x \mathrm{EUS}(S \cup \{x\})$. The submodularity of EUS implies that marginal benefits do not increase [82]. Specifically, let S_i and S_j be the sets of size i and j, respectively, be generated by greedy optimization at steps i and j—if $i \leq j$ then $\rho_{S_i}(x) \geq \rho_{S_j}(x)$. Therefore the values ρ can be stored and used as upper bounds on the marginal benefit of adding items to the set.

The greedy algorithm using lazy evaluation (see, e.g., Minoux [93]) proceeds as follows:

- The elements $x \in X$ are maintained in a list sorted by their marginal gain $\rho(x)$ relative to the current partial set. Initially, this marginal benefit $\rho_{\varnothing}(x)$ is simply its expected utility $\mathrm{EU}(x)$.
- At each round i with current set S_i, we compute new marginal gain values ρ_{S_i} w.r.t. S_i starting with elements $x \notin S_{i-1}$ with highest $\rho_{S_{i-1}}(x)$ and track the best value so far and its associated item y (note that each gain computation requires computation of EUS). If at some point the prior stored gain $\rho_{S_{i-1}}(x)$ of the current item is less than the current best gain $\rho_{S_i}(y)$, then we do not evaluate the updated gain $\rho_{S_i}(x)$, as it is necessarily less than than its prior value, and therefore less than the gain of y, i.e., $\rho_{S_i}(x) \leq \rho_{S_{i-1}}(x) < \rho_{S_i}(y)$. This holds for all items ranked lower than x w.r.t. prior gain $\rho_{S_{i-1}}$, so we stop the search and add item y to the set S as its ith element. Item y is removed from the list, which is resorted for the next iteration.
- We repeat until we obtain a set of size k.

In practice, often very few evaluations of EUS will be required at each iteration of greedy set construction (apart from the first round at which expected utilities for all candidate items is computed).

Query Iteration. The T transformation (Definition 19) gives rise to a natural heuristic method for computing good query/recommendation sets. This method, *query iteration (QI)*, starts with an initial set S, and locally optimizes S by repeatedly applying operator $T(S)$ until $\mathrm{EUS}(T(S);\theta)$=$\mathrm{EUS}(S;\theta)$. QI is sensitive to the initial set S—different initial sets can lead to different fixed points. We consider several initialization strategies:

- *Random:* randomly choose k options.
- *Sampling:* first add $x^*(\theta)$, the item with highest expected utility, to S. Then add the remaining $k-1$ items $x_i, 2 \leq i \leq k$ as follows : sample utility/weight vector $w^i \sim P(w;\theta)$, and add the optimal item $x_i = \arg\max_{y \in X} u(y; w^i)$ to S, enforcing, via rejection and resampling, that x_i is strictly better than all $x_1, \ldots, x_{i-1}$ under w^i.[7]
- *Greedy:* initialize with the set S_g obtained by simple greedy optimization of S (see above).

We can bound the performance of QI relative to optimal query/recommendation sets under both R_{NL} and R_C. If QI is initialized with S_g, performance is no worse than greedy optimization. If initialized with an arbitrary set, by submodularity we have:

$$\mathrm{EU}^*(\theta) \leq \mathrm{EUS}^*(\theta) \leq k\,\mathrm{EU}^*(\theta).$$

The condition $S \in \tau(S)$ implies $\mathrm{EUS}(S;\theta) = \mathrm{EPU}(S;\theta)$. Also note that, for any set Q, $\mathrm{EPU}(Q;\theta) \geq \mathrm{EU}^*(\theta)$. Thus, $\mathrm{EUS}(S;\theta) \geq \frac{1}{k}\mathrm{EUS}^*(\theta)$. This means for comparison queries ($|S| = 2$), QI achieves at least 50% of the optimal recommendation set value. This bound is tight and corresponds to the *singleton degenerate set* $S_d = \{x^*(\theta)\}$. This solution is problematic since $T(S_d) = S_d$ and has EVOI of zero. Note that S_d is the only singleton that is a fixed point:

Observation 40 *S_d is the only set with cardinality one that is a fixed point $S_d \in \tau(S_d)$.*

Proof. Applying T to any singleton is essentially a standard maximization of expected utility given the current belief, yielding S_d. □

By contrast, under R_{NL}, QI with sampling initialization avoids this fixed point provably by construction, always leading to a query set with positive EVOI. In fact, to avoid S_d, the initialization set S^0 is required to be such that $\mathrm{EUS}(S^0;\theta) > \mathrm{EU}^*(\theta)$. Note that $\mathrm{EUS}(S_d;\theta) = \mathrm{EU}^*(\theta)$ so query iteration (Lemma 23 and Corollary 24) guarantees that

$$\mathrm{EUS}(T(S^0);\theta) \geq \mathrm{EUS}(S^0;\theta) > \mathrm{EU}^*(\theta) = \mathrm{EUS}(S_d;\theta).$$

Thus, this optimization can never be trapped in the degenerate solution. Sampling initialization, in particular, meets this condition: the sampling procedure is such that each item contributes positively to EUS under noiseless response.

[7]Note that this may impossible if fewer than k items are optimal over all $w \in W$. Moreover, the procedure may be slow if, when adding the ith item, a prior item $x_1, \ldots, x_{k-1}$ is optimal for almost all $w \in W$. In practice, one can set a maximum number of attempts, and return a set of lower cardinality if all attempts fail.

	Asymptotic running times	
	General utility model	Linear utility model
EPU optimization	$O(n^{k+1}kl)$	$O(n^{k+1}k)$
EUS optimization	$O(n^k kl)$	$O(n^k k)$
Greedy EUS optimization	$O(nk^2 l)$	$O(nk^2)$
Query iteration	$O(nkl)$	$O(nk)$

Table 7: Summary of running times of various set optimization methods.

Observation 41 *Query iteration with sampling initialization converges to a non-degenerate set.*

The worst-case running time of an iteration of QI is $O(nkl)$. Specifically, it is linear in the number of options, similar to greedy optimization. However, in practice, it is much faster than greedy since typically $k \ll l$. While we have no theoretical results that limit the number of iterations required by QI to converge, in practice, a fixed point is reached in very quickly, as we demonstrate in our experimental results below.

Efficient computation in linear utility models. In a linear utility model, expected utility can be determined by computing the expectation of the utility parameters and treating these expectations as if they represent the true utility function. When $u(x; w) = w \cdot x$, where w is a vector of weights and x is a vector of numerical features, we have:

$$\mathrm{EU}(x; \theta) = \int_W u(x; w) P(w; \theta)\, dw = \int_W w{\cdot}x\, P(w; \theta)\, dw = x{\cdot} \int_W w P(w; \theta)\, dw = \mathbb{E}_\theta[w]{\cdot}x. \tag{85}$$

This simplifies a number of computations. Denoting $\bar{w} = E_\theta[w] = \int_W wp(w; \theta) dw$, the expected utility of an option x can be computed with a dot product, $\mathrm{EU}(x; \theta) = \bar{w} \cdot x$. Similarly, for computing EUS of a set S, we can substitute $\mathrm{EU}(x; \theta|S \rightsquigarrow x)$ with $\mathbb{E}_{\theta|S\rightsquigarrow x}[w] \cdot x$ (where $\mathbb{E}_{\theta|S\rightsquigarrow x}[w]$ is the expectation of the weight parameters with respect to posterior $\theta|S \rightsquigarrow x$), giving:

$$\mathrm{EUS}_R(S; \theta) = \sum_{x \in S} P_R(S \rightsquigarrow x; \theta)\, \mathrm{EU}(x; \theta|S \rightsquigarrow x) = \sum_{x \in S} P_R(S \rightsquigarrow x; \theta) \mathbb{E}_{\theta|S\rightsquigarrow x}[w]{\cdot}x. \tag{86}$$

This has implications for the running times of our proposed algorithms for generating sets. With a linear model, evaluation of EPU for a set S becomes $O(nk + lk) = O(nk)$, since we are interest in the running time with respect to n (viewing k and l as constant), and naïve optimization of EPU becomes $O(n^{k+1}k)$. Greedy maximization for a linear model with respect to EUS becomes even faster, $O(nk^2)$, still linear with respect to n but with a smaller constant. Query iteration with a linear model has running time $O(nk)$. Table 7 summarizes the running times of our proposed algorithms.

5. Empirical Evaluation

In this section, we evaluate the various methods developed in Sections 3.5 and 4.7 for computing recommendation and query sets, within both the regret-based and Bayesian frameworks. We begin in Section 5.1 by describing our datasets and methodology. We present experiments with optimization and elicitation using our minimax-regret-based algorithms in Section 5.2, while Section 5.3 describes our results using our Bayesian optimization and elicitation techniques.

5.1. Datasets and Methodology

We test our algorithms by simulating the responses and choices of users with randomly-generated utility functions over a variety of domains. We use four different datasets, three of which are small databases containing from 81 to 506 multi-attribute items, and the fourth of which is a family of randomly generated constraint-based configuration problems.

Synthetic A small synthetic dataset of 81 items, consisting of the full Cartesian product of four categorical features, each with domain size three. This small dataset is useful because it allows the fast computation of optimal recommendation sets using exact algorithms (something not feasible with larger datasets). This serves as a valuable benchmark for assessing the quality of the recommendation/query sets generated by our approximation algorithms.

Rental A database of 187 rental accommodation options, drawn from a real university student rental database. Each option is characterized by ten features, such as price, size, distance to university, etc. Some feature domains are numeric, both real-valued and integer (e.g., price, number of rooms); others are categorical with domains of size 2–6 (e.g., neighborhood). We assume an additive utility model in which utility over numeric attributes is linear. Specifically, numeric features have a single (unknown) utility weight, while local utility over the categorical domains are specified by a vector of weights on $[0, 1]$, one per domain value. The least preferred domain value has local value 0, and we assume that the local preference ordering of the domain values is known (e.g., that "shared apartment" is less preferred than "studio apartment"), although it can differ among users.

Boston The Boston housing dataset[8] contains 506 items with one binary and 13 real-valued features. Real-valued features are pre-processed to lie within a common scale. We assume that user utilities are linear. We make no assumption about whether high or low values are preferred, thus we allow weights to fall in the interval $[-1, 1]$.

Configuration We generate random configuration problems as follows. We assume 15 binary attributes and assume domain value 1 is preferred to 0 for each. For a specific problem instances, we generate 30 random feasibility constraints, each obtained by sampling 2 to 6 attributes and imposing the feasibility constraint that at least one of the attributes must take value 0. On average, random configuration problems generated this way have more than 5000 feasible solutions (i.e., possible items that can be recommended to or chosen by a user). We generate ten random configuration problems, and test our optimization/elicitation methods on each configuration problem using ten random users (utility functions), thus averaging results over 100 problem-user instances. We only test regret-based elicitation on these configuration problems.

We test our algorithms by simulating the interaction with *random users*, that is, users with random utility functions (i.e., weight vectors) suitable for the dataset/domain in question, drawn from some distribution. Users make choices from a recommendation set or query set based on this underlying utility function, which is initially unknown, but gradually estimated by the algorithm. In the Bayesian setting, user choice or query response may be subject to noise using one of the noise models analyzed above. Results are averaged over 20–50 random users/utility functions.

Since our algorithms generate sets S that can be used as both recommendations and as set queries, we test our set-construction techniques in a way that evaluates them simultaneously as recommendation and elicitation strategies. Specifically, at each round t of interaction with a simulated user, a set is presented to the user, the user responds with a choice, and the system updates its belief state given that response and moves on to round $t + 1$. We measure (using actual utility, expected utility, or max regret) the quality of the best recommended item at each round. This provides an "anytime" evaluation of the algorithm in question that can be interpreted in two ways. (1) We can view this as a process that terminates after T rounds, with our results showing final decision quality for various values of T. (2) We can view this as an anytime method in which the system, or the user, decides to terminate when they find the quality of the selected item to be satisfactory (perhaps relative to the cost of further interaction).

We compare different algorithms on specific subsets of the domains above and compare them using recommendation/query sets of multiple sizes k. We also examine the computation time of various algorithms in the more computationally intensive Bayesian setting.

We consider several of our regret-based algorithms in the experiments below, including (see Section 3.5.2): setwise chain of adversaries (SCAS), which is equivalent to the current-solution strategy (CSS) for sets of size $k = 2$; and query iteration (QI) using SCAS as a seed (dubbed QI-SCAS). We also use the set with true setwise minimax regret (SMMR) for small sets ($k = 2$). Finally, we consider the *top-k strategy* (dubbed regret-top-k), which selects the k items with least max regret, and a *random strategy* which recommends or queries with k random items.

[8]`http://lib.stat.cmu.edu/datasets/boston`

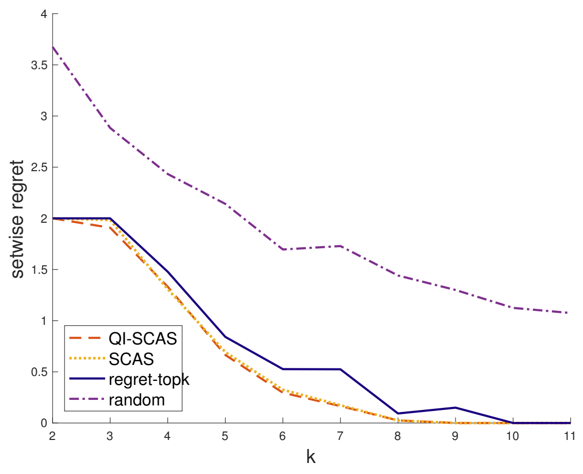


Figure 10: Setwise max-regret as a function of size k (synthetic dataset, 20 runs).

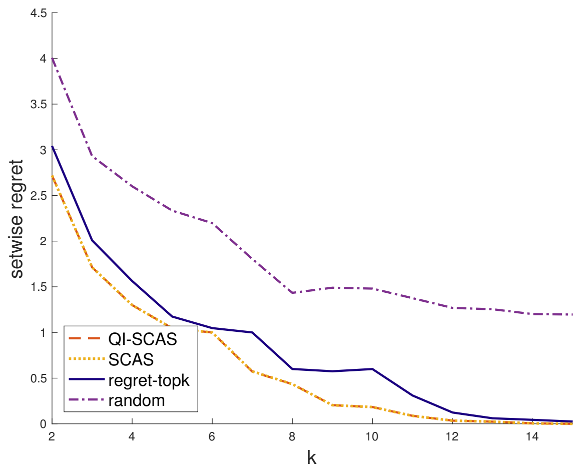


Figure 11: Setwise max-regret as a function of size k (rental dataset, 20 runs).

In the Bayesian experiments, we consider the following algorithms (see Section 4.7): greedy optimization of EUS with lazy evaluation (GreedyEUS), query iteration with sampling (dubbed QIsampling), and query iteration with random initialization (dubbed QIrandom). In some experiments, we also consider the *top-k strategy* (dubbed EUtop-k), which selects the k items with highest expected utility. In small problems, we also consider naïve maximization of EUS (shown as EUSopt in the figures), which computes the true optimal recommendation set. We also include the random strategy as baseline.

5.2. Regret-based Recommendation and Elicitation

In this subsection, we assess the performance of our minimax-regret-based methods.

Setwise max regret and set size. Prior to exploring anytime elicitation performance, we first compare the ability of different algorithms to generate good recommendation sets when they are given the same preferences as input. The experimental set up is similar to that considered by [100]. We also explore the degree to which setwise max regret SMR changes as a function of recommendation/query set size k. (By Observation 2, of course, we know larger sets have lower SMR.) Figures 10 (synthetic data set) and 11 (rental dataset) plot SMR for different values of k for four algorithms: SCAS, QI-SCAS, top-k and Random. Evaluation of SMR is based on the recommendation set generated after the user has responded to five queries.

We see that, even when considering the same input preferences, there can be large differences in quality of the recommendation sets. While top-k generates good recommendation sets, these are generally suboptimal. Not surprisingly, SMR drops with increasing set size for each method, but it drops more quickly with SCAS and QI-SCAS than with the others. Furthermore, both SCAS and QI-SCAS yield lower SMR than top-k across the entire range of set sizes, but perform similarly in these two domains. Random performs quite poorly.

Incremental elicitation in database problems. We next evaluate the elicitation and recommendation performance of our regret-based setwise strategies by examining the reduction in minimax regret as elicitation proceeds, specifically, as a function of the query round t. The (single-item) minimax regret at round t corresponds to the final recommendation in the case that elicitation stops at round t; in other words, we assume the system provides a final recommendation when elicitation is terminated. We perform this evaluation on all three datasets and vary the number of options k. Apart from minimax regret, we also measure *true regret*, that is, the user's true utility for the minimax-optimal recommendation relative to the *true optimal* item with respect to the user's true utility.

Minimax regret is an upper bound on true regret; indeed, the recommender may recommend the optimal option to a user long before it can "prove" that the recommended item is optimal. Of course, the recommender cannot actually determine this true regret (unless max regret is zero), but we illustrate it here to give an indication of the performance

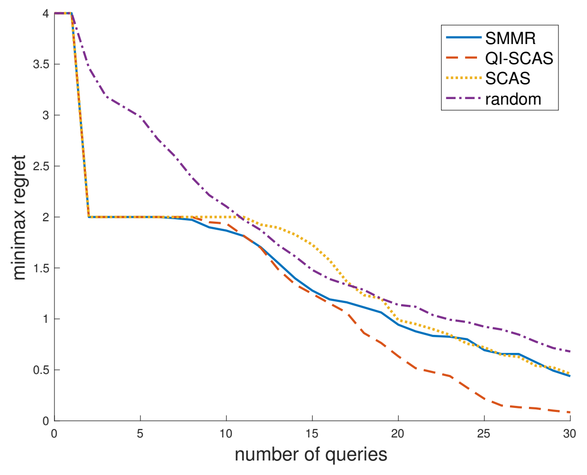


Figure 12: Minimax regret reduction ($k = 2$, synthetic dataset, 30 runs).

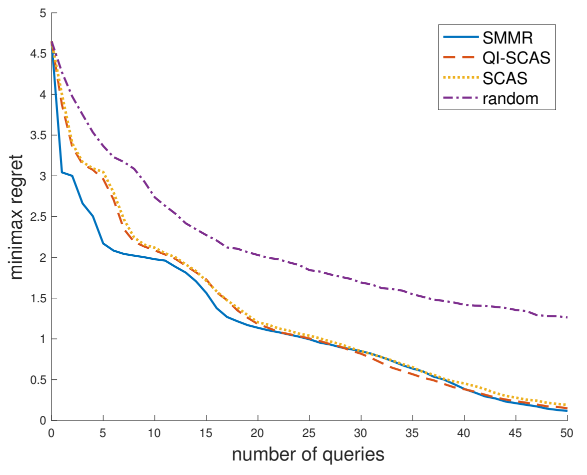


Figure 13: Minimax regret reduction ($k = 2$, rental dataset, 30 runs).

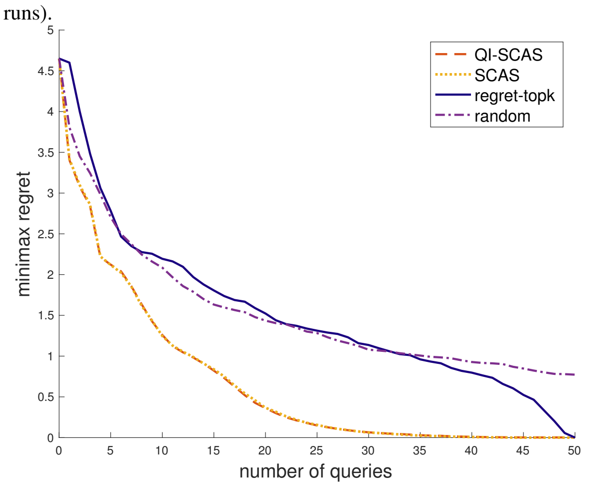


Figure 14: Minimax regret reduction ($k = 3$, rental dataset, 50 runs).

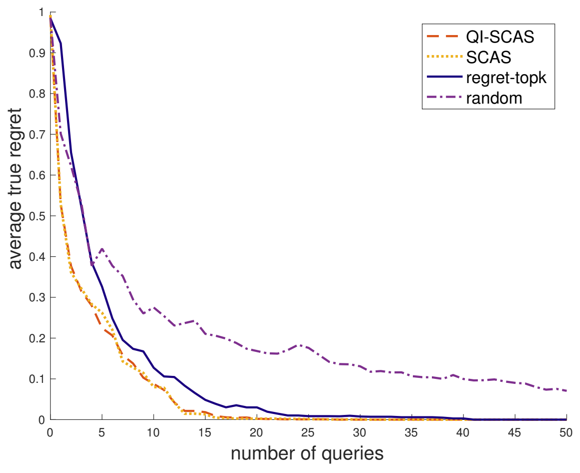


Figure 15: True regret ($k = 3$, rental dataset, 50 runs).

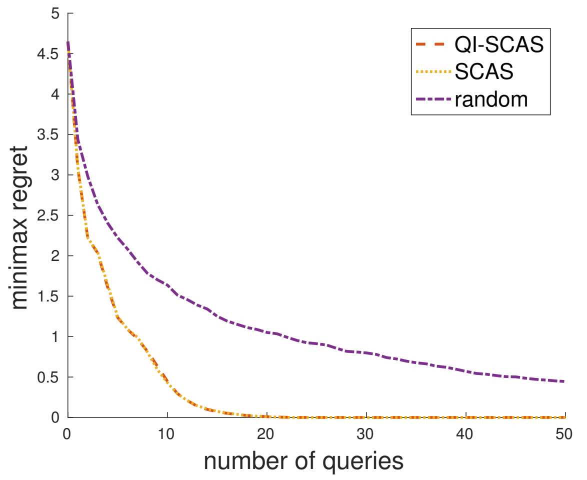


Figure 16: Minimax regret ($k = 5$, rental dataset, 50 runs).

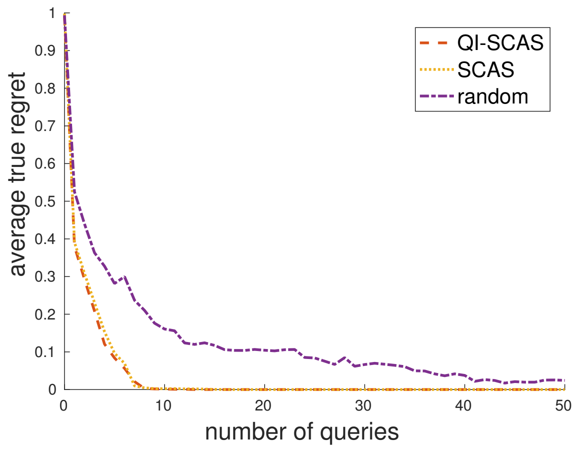


Figure 17: True regret ($k = 5$, rental dataset, 50 runs).

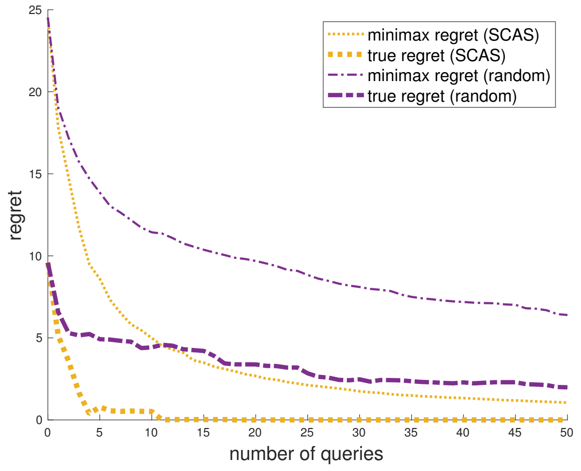


Figure 18: Average true regret vs. minimax regret (SCAS, $k = 3$, Boston dataset, 30 runs).

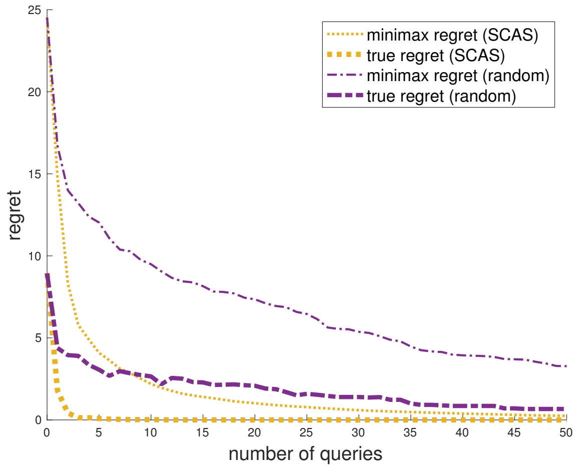


Figure 19: Average true regret vs. minimax regret (SCAS, $k = 6$, Boston dataset, 20 runs).

of each algorithm relative to a user's true preferences. In the following experiments, the parameters of the true utility function are sampled uniformly at random (we sample each parameter i.i.d. between the lower and upper bound).

We first consider the synthetic dataset of 81 items and recommendation sets of size $k = 2$—the small set size permits exact computation of SMMR. Figure 12 plots minimax regret of as function of the elicitation round, comparing SMMR, QI-SCAS, SCAS/CSS[9] and random. While SMMR is effective up to the 20th query, it gradually trails QI. We conjecture that the "dense" nature of this specific dataset (where each combination of feature values is feasible) makes it somewhat harder for SMMR to generate good sets, since there may often be ties (i.e., sets with the same SMR), making it difficult to discriminate among more or less informative sets. Interestingly, QI performs somewhat better than both SCAS/CSS and SMMR: by optimizing each element of the S-cover, QI improves not only WR, but apparently also the *a posteriori* regret given any answer to a query.

Figure 13 shows the same evaluation using the Rental dataset. While SMMR is optimal, both SCAS and QI offer very good approximations, and are much more computationally efficient. For this reason, we focus on SCAS and QI in the remaining experiments.

We next compare the QI-SCAS, SCAS and Random on the Rental dataset using larger set sizes $k = 3$ and $k = 5$, and also evaluate top-k for $k = 3$. For $k = 3$, Figures 14 and 15 plot minimax regret and true regret; and for $k = 5$, Figures 16 and 17 do the same. Both regret-based query strategies, Q-SCAS and SCAS, perform similarly. Each reduces max regret very quickly, reaching zero minimax regret in about 35 queries on average (resp., 20) when $k = 3$ (resp., $k = 5$). True regret reaches zero significantly faster, after roughly 25 (resp., 10) queries. In the $k = 3$ case, top-k does a relatively poor job of reducing minimax regret, performing no better than Random for the first 35 rounds and requiring 50 queries on average to reach zero minimax regret. Its recommendations do have relatively low true regret—though this cannot be known to the recommender—but it still lags the regret-based methods on this metric as well. The benefits of reasoning explicitly about the "joint value" of a set (either as a recommendation or a query) rather than focusing on the individual items is evident in this comparison with top-k.

Finally, we test SCAS and random on the Boston Housing dataset using $k = 3$ (see Figure 18) and $k = 6$ (see Figure 19). The anytime performance of the algorithm with respect to minimax regret and true regret is similar to the Rental domain.

Elicitation in configuration problems. We next examine regret-based elicitation strategies in the configuration setting. We compare SMMR (using the mixed integer program of Equations 24-31), SCAS and QI-SCAS with Random. Figures 20 and 21 show the minimax regret after each round of elicitation for set sizes $k = 2$ and $k = 5$, respectively. SMMR is tested only when $k = 2$ since it is too computationally intensive when $k = 5$ (see results on computation

[9]When $k = 2$, CSS and SCAS are identical.

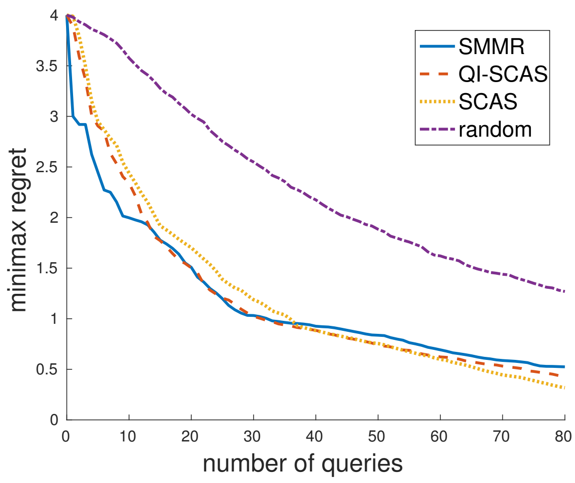


Figure 20: Max regret reduction in configuration problems ($k = 2$; 100 total runs)

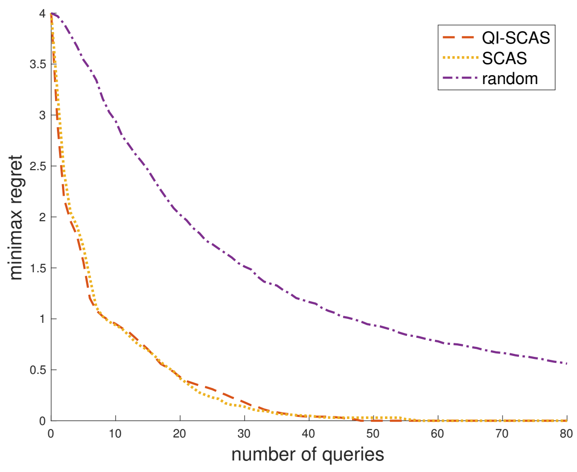


Figure 21: Max regret reduction in configuration problems ($k = 5$; 100 total runs)

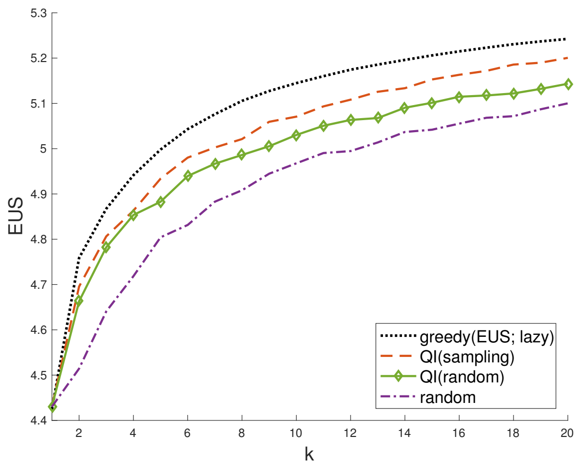


Figure 22: EUS wrt set size (rental dataset, 30 runs, R_{NL})

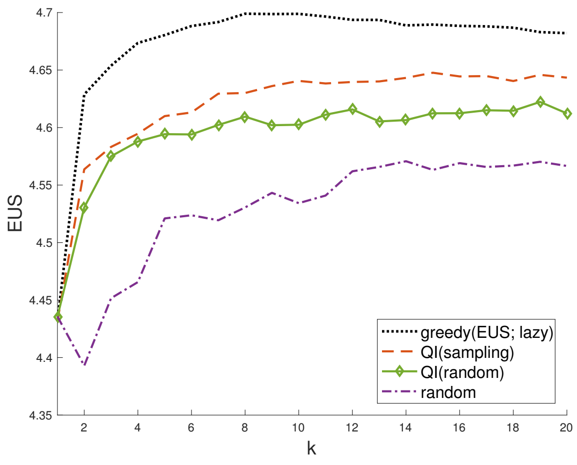


Figure 23: EUS wrt set size (rental dataset, 30 runs, R_L with $\gamma = 1.5$)

time in Table 8). When $k = 2$, SMMR outperforms QI-SCAS and SCAS over the first 30 rounds, at which point the two heuristics both match, then improve, minimax regret relative to SMMR. When $k = 5$, QI-SCAS and SCAS have very close performance.

Computation time. We compare the computation times of the different optimization methods in Table 8. In the case of configuration problems, we use the integer programming formulation developed in Section 3.5.1. Unsurprisingly, the optimal method SMMR is only feasible for very small k; in database problems, $k = 3$ is infeasible, and even for configuration problems $k = 3$ is challenging enough to make SMMR impractical for interactive elicitation. Both SCAS and QI-SCAS are quite practical, with SCAS being fast enough across all problems to support the real-time, interactive optimization required to drive preference elicitation. Given the negligible differences in recommendation set and query quality between SCAS and QI-SCAS, this would suggest that SCAS is most promising, at least in the domains tested here.

5.3. Bayesian Recommendation and Elicitation

In this subsection, we assess the performance of our Bayesian methods.

		Computation Times		
dataset	k (set size)	SMMR	QI	SCAS
synthetic	2	19.47	0.78	0.31
synthetic	3	-	1.92	0.40
synthetic	5	-	5.49	0.36
synthetic	7	-	12.53	0.48
rental	2	37.52	1.93	0.69
rental	3	-	1.32	0.34
rental	5	-	1.66	0.46
rental	7	-	2.68	0.69
boston	2	277.33	2.81	0.72
boston	3	-	2.99	1.10
boston	5	-	20.40	2.12
boston	7	-	11.34	3.35
configuration	2	0.75	0.40	0.10
configuration	3	39.07	0.64	0.13
configuration	5	742.07	1.15	0.20
configuration	7	-	1.67	0.36

Table 8: Average computation times (in seconds) of different strategies for generating recommendation sets in the regret-based framework.

EUS computation and set size. As in the regret-based setting, we first evaluate the ability of our algorithms to generate good recommendation sets. For this task, we compare the expected utility of selection EUS of the recommended sets when considering the same input preferences for all methods (and therefore the same belief θ) We analyse the impact of the size of the recommendation set, varying k from 2 to 7, on EUS in the Rental dataset (187 items). Figure 22 shows EUS as a function of set size, assuming a noiseless response model, for several of the Bayesian recommendation heuristics as well as the Random strategy. EUS is shown for the recommendation sets generated after the user has provided five preference statements. Figure 23 shows the same results when responses have logistic noise with $\gamma = 0.33$. EUS is not monotone in k with such noisy responses: indeed, notice that EUS of the greedy method reaches its maximum value when $k = 8$ and slowly decreases after that. Similarly, EUS obtained by the other methods quickly improves for small values of k but then stalls when k is greater than 10 to 12. Using randomly generated queries, EUS_L when $k = 2$ is on average less than the expected utility of the current best recommendation; this is not surprising, since Random will often add "uninteresting" elements with low utility to the recommendation set—with response noise, these are selected by the user with some non-trivial probability, lowering the expected utility of the set.

Incremental elicitation with noiseless responses. We next compare the elicitation/recommendation strategies on recommendation problems with random user utility functions assuming noiseless user response. We use two methods to sample utility functions: 1) uniformly distributed parameters; and 2) normally distributed utility parameters. In each case, each parameter is sampled i.i.d. We assume that the Bayesian recommender system is given the correct prior, hence, the initial belief θ corresponds to the sampling distribution.

Bayesian inference is realized using Monte Carlo with importance sampling, with particle weights determined by applying the response model to observed user responses. (We describe the method in general, which applies to noisy response models as well, as examined below.) To overcome the problem of particle degeneration (i.e., the fact that particles tend to eventually evolve to have low or zero weight), we use slice sampling [95] to regenerate

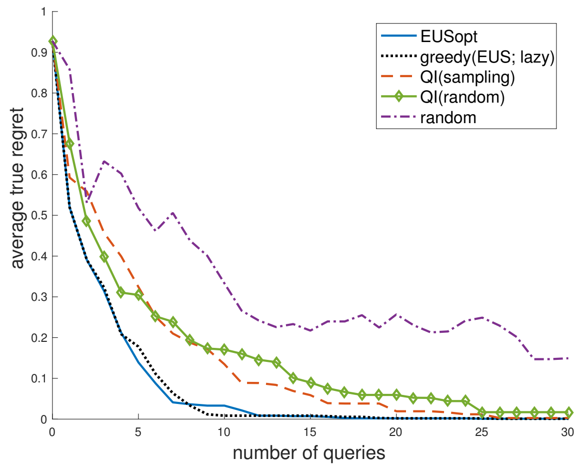


Figure 24: Average true regret (rental dataset, $k = 2$, R_{NL}, uniformly distributed utilities; 30 runs).

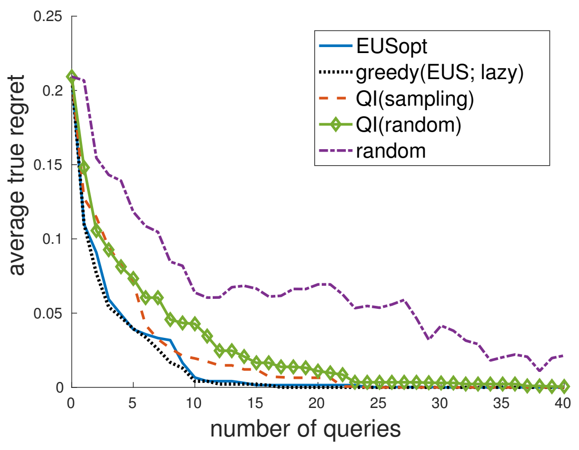


Figure 25: Average true regret (rental dataset, $k = 2$, R_{NL}, normally distributed utilities; 30 runs).

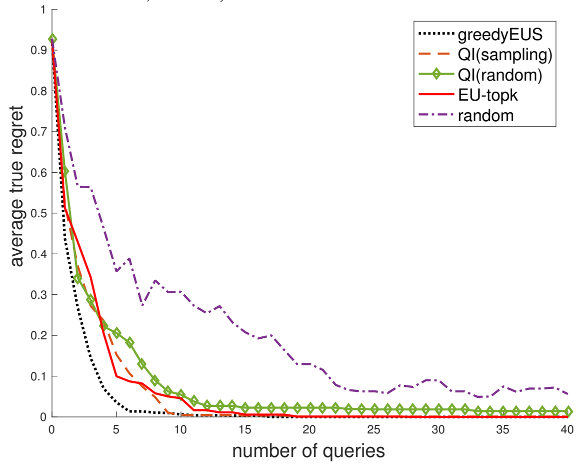


Figure 26: Average true regret (rental dataset, $k = 3$, 30 runs, R_{NL}, uniformly distributed utilities; 30 runs).

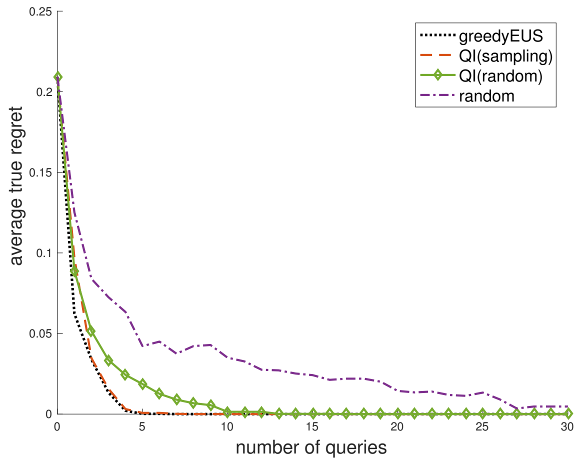


Figure 27: Average true regret (rental dataset, $k = 5$, R_{NL}, normally distributed utilities; 30 runs).

dataset	γ (noise)	average-Δ	min-Δ	max-Δ	bound $\hat{\Delta}$
synthetic	1	0.1644	0.1256	0.1827	0.2785
synthetic	10	0.0092	0.0074	0.0129	0.0278
synthetic	30	0.0011	0.0008	0.0017	0.0093
rental apt	1	0.1692	0.0052	0.2393	0.2785
rental apt	10	0.0087	0.0019	0.0238	0.0278
rental apt	30	0.0011	0.0001	0.0074	0.0093
boston housing	1	0.1653	0.0300	0.2184	0.2785
boston housing	10	0.0027	0.0007	0.0208	0.0278
boston housing	30	0.0003	0.0001	0.0061	0.0093

Table 9: Actual loss values of pairs (i.e. $k = 2$) compared to the bound on maximum loss.

particles w.r.t. the response-updated belief state θ whenever the effective number of samples drops significantly (50000 particles were used in our simulations). Figures 24 (with uniformly distributed utility parameters) and Figure 25 (normally distributed utility parameters) show the average loss of our strategies in Rental dataset with $k = 2$. Greedy optimization performs almost as well as exact optimization, and the optimal item is found in roughly 10–15 queries. Query iteration performs reasonably well, though not as well as Greedy. Initializing QI with sampling provides slightly better performance than using random seeds.

In Figure 26 we show the result of incremental elicitation with choice queries of size three ($k = 3$). This experiment includes EU-topk, the non adaptive strategy that selects the k items with highest expected utility. EU-topk performs poorly in the early rounds but gradually catches up and performs on par with QI.

Figure 27 shows experimental results with $k = 5$ with normally distributed utility parameters. In this setting, QIsampling performs particularly well and is almost as good as GreedyEUS; we noticed a similar pattern in other tests (unreported) with normally distributed parameters. We attribute this to fact that, with an informative prior, the sampling initialization can focus the search in more promising query sets.

Bound of the logistic noise response model. Using the Rental dataset, we compare the actual values of $\text{EUS}_{\text{NL}}(S) - \text{EUS}_L(S)$, the loss due to logistic noise, with the worst-case bound of Theorems 30 and 31. Table 9 shows the average, minimum and maximum loss over all pairs of items in the dataset. The last column displays the bound $\hat{\Delta}$ provided by Theorem 31. Apart from verifying our theoretical results, the table demonstrates that the loss due to response noise is often much lower than the bound in practice.

Elicitation with logistic noise. We now examine elicitation performance under logistic response noise, and assess the impact of the temperature parameter γ, specifically, testing $\gamma = 20$ ("low" noise) and $\gamma = 5$ ("high" noise). We test Greedy, QI with both sampling and random initial seeds, and Random. In these experiments. Greedy is optimized "assuming" noiseless responses, but is evaluated using the true response model R_L (other experiments, not reported, show no advantage in optimizing EUS_L).

Figures 28 and 29 show results on the Rental dataset. We see that, in both cases, QI with sampling performs significantly better than QIrandom and almost as well as GreedyEUS. "High" noise delays convergence of elicitation in finding the true best item to some extent: GreedyEUS converges to the optimal item in five rounds when $\gamma = 20$, but needs about 12 rounds when $\gamma = 5$; similarly, QIsampling needs 8 and 10 rounds, respectively.

The Boston housing dataset is examined in Figures 30 (low noise) and 31 (high noise). While all query strategies are effective in the low-noise scenario (average loss is zero or negligible after 10 rounds), in the high-noise scenario we see that the average loss of GreedyEUS, after having decreased considerably, fails to reach 0 before the end of the simulation (upon inspection, we notice that GreedyEUS usually converges to true best items, but stalls with a suboptimal item on a small number of runs). In this setting, however, query iteration performs quite well, in particular QIsampling.

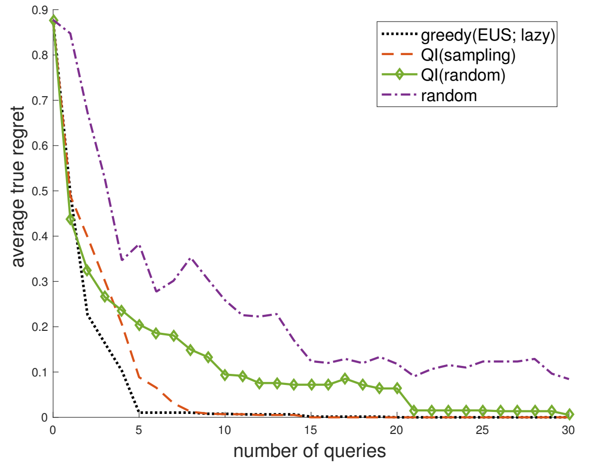


Figure 28: Average true regret in "low" noise scenario (rental dataset, $k=3$, R_L, $\gamma = 20$, uniformly distributed utilities; 20 runs).

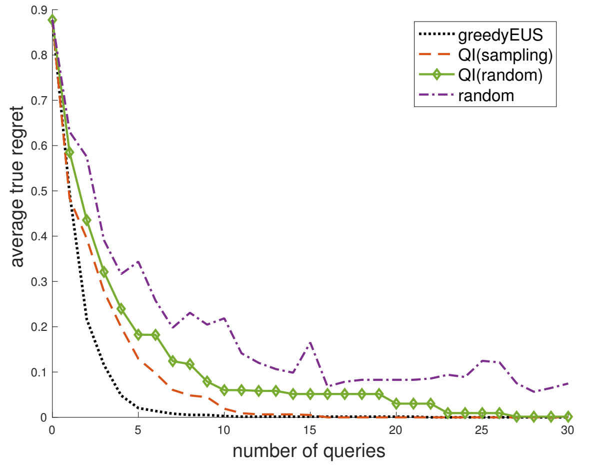


Figure 29: Average true regret in "high" noise scenario (rental dataset, $k=3$, R_L, $\gamma = 5$, uniformly distributed utilities; 20 runs).

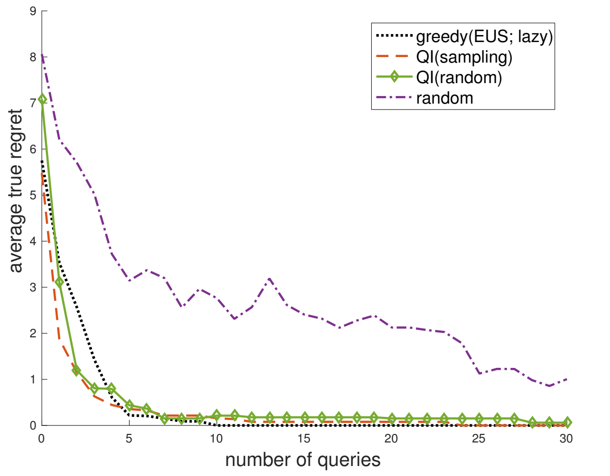


Figure 30: Average true regret in "low" noise scenario (Boston dataset, $k=2$, R_L, $\gamma = 20$, uniform utilities; 20 runs).

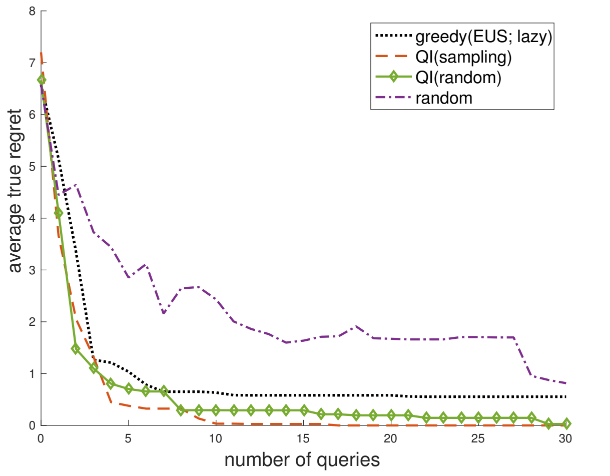


Figure 31: Average true regret in "high" noise scenario (Boston dataset, $k=2$, R_L, $\gamma = 5$, uniform utilities; 20 runs).

Overall, these experiments show that (greedy or approximate) maximization of EUS is often able to find good query sets even in the case of noisy responses. However, the lack of convergence of GreedyEUS in the last experiment shows that the theoretical guarantees of (greedy) EUS maximization may not be sufficient in practice. Given the strong performance of QIsampling in noisy settings, we think that an investigation of query iteration and similar strategies may be a promising direction for further research.

Computation times. Finally, we compare recommendation/query set optimization times of the different Bayesian methods on the three database-oriented datasets in Table 10. Among our strategies, QI is certainly most efficient computationally, and is best suited to large outcome spaces. Interestingly, QI is sometimes faster when initialized using sampling rather than random seeds because it needs fewer iterations to converge (due to the rejection involved in the random initialization). Note that this table shows only the computation time for generation of the recommendation/query sets themselves, and does not include the time needed for Bayesian inference (updating the posterior given a user response). In our simulations, Bayesian updates with importance sampling resampling performs well, requiring less than a second in all datasets. However, in our implementation of preference elicitation with simulated users, every few steps (typically 4–8) we need to sample additional particles from the updated posterior in order to

		Computation Times			
dataset	*k* (set size)	EUSopt	greedyEUS	QI(sampling)	QI(random)
synthetic	2	1.22	0.03	0.01	0.01
synthetic	3	36.21	0.03	0.01	0.01
synthetic	5	289.21	0.04	0.01	0.02
synthetic	7	-	0.05	0.01	0.01
rental	2	2.55	0.06	0.01	0.01
rental	3	97.95	0.07	0.01	0.01
rental	5	-	0.08	0.01	0.02
rental	7	-	0.09	0.01	0.02
boston	2	41.26	0.06	0.01	0.01
boston	3	-	0.07	0.01	0.01
boston	5	-	0.10	0.01	0.01
boston	7	-	0.13	0.03	0.01

Table 10: Average computation times (in seconds) of different strategies for generating recommendation sets in the Bayesian framework.

avoid degeneracy. This is significantly slower (with 50000 particles the time needed is around 30–60s., depending on the dataset).

6. Future Directions, Extensions and Related Work

In this work we consider the value of a set with respect to its capacity to "cover" the uncertainty associated with a partially known user utility function. The underlying assumption is that the user is looking for a single item to consume, purchase or use, and the quality of a recommendation set is related to the odds that at least one item in the set has high utility for the user. It is important to note that other models consider recommendation sets under a different semantics, for example, dealing with the problem of recommending a set that accounts for positive or negative synergies between items [44, 60] .

One key advantage of our framework is that it supports anytime recommendations—our techniques can be used to leave the decision to accept a current recommendation or engage in further elicitation in the hands of the user. Indeed one of the motivations of this work is the need for interactive recommender systems that allow some form of user-directed exploration of the item space. A possible application is to augment *example-critiquing tools* [134], with principled utility-elicitation techniques. Of course, it is challenging to develop precise mathematical models for open-ended user responses, such as critiquing a displayed example. Another interesting area of investigation consists in elicitation methods for qualitative decision problems [15, 41].

One could consider alternative criteria, other than probabilistic methods or minimax regret as considered here, for aggregating the uncertainty over the utility function. Maximin is one such criterion, in which one recommends the item with greatest utility in the worst case [136, 112]. The use of maximin in the context of adaptive utility elicitation is studied by [135], who present a similar connection between the optimality of a query and the optimality of a recommendation set. However, maximin as a decision criterion is such that myopic optimization is often unable to differentiate queries: since maximin is inherently pessimistic, in many cases no single choice query is able to improve maximin utility in one step, and heuristic tie-breaking schemes are needed. This often renders incremental elicitation ineffective.

While we focused on the notion of helping a user/decision maker identify her most preferred item from a wide set of possibilities, our techniques can also be applied to collaborative-filtering based recommender systems. There active elicitation might take the form of asking the user queries about the her ratings (as a proxy for utility) of different items (for instance, movies that the user is likely to have seen) [83]. In particular, EVOI-based queries may perform better than techniques presented in [48] to select "rating queries" (asking the user to rate a set of movies); we refer to [23] for one use of EVOI for elicitation in collaborative filtering domains.

In sequential problems, utility uncertainty takes the form of partially specified reward values in Markov decision processes (MDPs) or reinforcement learning, where the system recommends a policy π, mapping states to actions. Akrour et al. [5] apply our EUS criterion to the problem of selecting the best pair of trajectories to present to the user for a comparison query. A noisy response model is considered where probability of error decreases linearly with an increasing utility gap between the two trajectories. Regan and Boutilier [101] use minimax regret to explicitly elicit MDP reward functions from users.

A recent active area of research is the automatic assessment of the weights of complex non-linear decision models, in particular, rank-dependent aggregators as the Choquet integral [56, 55]. Eliciting or learning a utility model based on the Choquet integral is challenging however because of the number of parameters that must be assessed, which is exponential in the number of attributes. In particular, technical problems arise due to the number of constraints required to assure monotonicity of the underlying capacity measure (a property deemed essential). The problem can be relaxed by making additional assumptions. For instance, Tehrani *et al.* [121] restrict attention to Choquet models determined by 2-additive capacities. Another approach limits attention to query types that are easy to handle—[12] provides efficient computation and elicitation techniques based on minimax regret, at the cost of restricting to comparison queries involving fictitious alternatives.

A number of important directions for future research remain. Further theoretical and practical investigation of local search strategies such as query iteration is important. Another direction is the development of strategies for Bayesian recommendation and elicitation in large-scale configuration problems, e.g., where outcomes are specified by a constraint satisfaction problem, and for sequential decision problems (such as MDPs with uncertain rewards). It is also interesting to study how to automatically learn user response models from preference data. We are interested in the problem of preference elicitation when users have specific risk attitudes [65, 98]. We also believe that there is potential value in elicitation strategies that combine probabilistic and regret-based models.

We now discuss connections to related works in machine learning. The last several years have seen a dramatic shift to the use of *deep neural network (DNN)* models across machine learning, and for recommender systems in particular. The move to DNNs has transformed recommender research and practice (e.g., recommendations for video on YouTube [42], movies on Netflix [53], and music on Spotify [70]), allowing complex non-linear user behavior to be modeled. Furthermore, multi-task models allow recommenders to predict not just user clicks, choices, or ratings, but *multiple* aspects of a user's response to a recommendation [129, 137, 42, 37] to predict user responses (e.g., content engagement, likes or shares) to generate, score and serve recommended items.

Our general results on the relationship between optimal set recommendations and optimal choice queries are independent of the "representation" of user preferences or choice behvaior—for example, some DNN models can be loosely interpreted as uncovering or constructing latent feature representations of items over which user utility (or response behavior) is defined. Our general algorithmic techniques (e.g., query iteration) are also independent of the particular form of the user utility model; but many of the concrete algorithms we propose assume, for example, linear utility over item attributes. An important direction for future research is to develop tractable algorithmic methods for computing key notions like setwise max regret, worst-case a posteriori regret, expected utility of selection, expected posterior utility, and expected value of information with DNN preference representations or utility models, and especially adding uncertainty representations to such models. One direction might be to treat DNN hidden layers as domain features, and treating the last layer as a linear utility model, as proposed in recent work on Bayesian exploration using DNNs [107].

A fair amount of work in recommenders has also begun to consider the impact that an entire sequence of recommendations has on a user, especially (though not exclusively) in content-based recommendation where users generally consume multiple recommended items over an extended period. As a consequence, the use of temporal models such as hidden Markov models and recurrent neural networks [105, 31, 62, 109, 120, 141], and long-term planning using *reinforcement learning (RL)* techniques (e.g., [116, 119, 51, 38]) is relevant. Very recently, work modeling the impact of *slates* of recommendations on RL methods has been considered [144, 36, 68], providing an interesting lever for engaging in direct preference elicitation. An important direction for future research is the integration of elicitation techniques like ours—which involve inherently sequential interaction with users—into recent RL methods.

Elicitation techniques bear a natural connection to exploration-based methods such as those based on *bandit* *algorithms* [14], since both aim to uncover (or sharpen their estimates of) the value of a set of alternatives (e.g., items). Bandit methods have been proposed for recommender systems by a number of authors (e.g., [75, 138]); but the nature of the interaction with users is generally different in bandit methods, which usually suggest items and receive direct (possibly stochastic) feedback of the true value of the item suggested (in bandit parlance, of the arm pulled). The algorithms are usually designed to minimize regret over the interaction horizon (or to identify the best arm) and do not address the theoretical or practical questions we consider regarding the connection between "queries" and "recommendations" as we do.

The recent literature on *dueling bandits* [4, 117] and related methods such as *Bayesian optimization* with comparison feedback [39, 16, 54], bears a stronger connection to our work. These models generalize bandits and Bayesian optimization by allowing pairwise feedback comparing two alternatives—which has higher value or is preferred—rather than providing direct "values" or utilities. (As mentioned earlier, a pairwise comparison is equivalent to a choice query of size $k = 2$.) Augmenting our analyses to study the regret and/or query complexity of our heuristic elicitation schemes is of tremendous interest; we refer to Dragone et al. [46] for excellent progress in that direction.

While our emphasis in studying setwise recommendations was its connections to elicitation, considerable attention has been paid to set-based recommendation in itself, especially work that explicitly tries to capture diversity in the set. Apart from work discussed earlier, recent work on diversity includes the use of determinantal point processes [140]. Work on off-policy correction for slate recommendations is relevant to learning models of user responses from data (e.g., [118, 36]).

More broadly, an interesting extension of our approach is the investigation of alternative user choice models, and how they might impact our results. Parametric and semi-parametric models offer the greatest potential for formal analysis, e.g., hierarchical extensions of the multinomial logit model, or *cascade models* [73, 43, 80] have proven popular in the machine learning community to explaining user interactions with ordered lists of search results, recommendations, search results, etc. (these are especially effective at capturing position bias). Recently, DNN, sequence and autoencoder models have been developed for explaining user choice behavior in recommendation sets or lists in a more general, non-parametric fashion [3, 9, 72]. Extending our analysis or empirical techniques to such general models would be especially fruitful.

7. Conclusions

Assessing user preferences, either directly or indirectly, is a crucial task in many AI applications. Explicit preference elicitation requires that one choose informative queries to quickly converge to good recommendations and limit the cognitive cost imposed on users. Among the many possible types of queries, choice queries ask a user to indicate which choice (item or product) is most preferred from a set of k options. Often a *set of recommendations* is presented to the user, as showing a diverse set of recommendations increases the odds of recommending at least one item with high utility to the user. Note that a set of items can also be presented to the user with the simultaneous goals of recommendation and of elicitation.

In this article, we have provided a novel analysis of set-based recommendations in recommender systems, and have shown how it is offers a tractable means of generating myopically optimal or near-optimal choice queries for preference elicitation. We examined two frameworks, one based on minimax regret and the other on Bayesian decision theory. These models offer a well-defined semantics for both set recommendation and choice queries. We developed strong theoretical connections between set-based recommendations and queries in both settings that are strongly analogous.

In the regret-based approach, we use minimax regret as a principled way to generate recommendations under strict uncertainty and formalized a natural extension of the itemwise minimax regret decision criterion to sets. We proved that optimal recommendations sets are also optimal query sets, when query sets are evaluated with respect to myopic worst-case regret reduction.

In the Bayesian model, we considered several user response models, showing that optimal recommendation sets are also optimal query sets, when queries are evaluated using expected value of information, under both noiseless and constant-noise user response models. Under the logistic/Luce-Sheppard response model, we showed that optimal recommendation sets are near-optimal query sets, both theoretically and experimentally. We stress that our results are general and do not depend on the specific form of the Bayesian prior, nor that of the utility function, though there is some dependence on the user response model. Our greedy strategies, which exploit the submodularity of the expected utility of selection from a set, perform very well in practice and have theoretical approximation guarantees. Finally our experimental results demonstrated that query iteration, a simple local search strategy, is especially well-suited to large decision spaces.

Naturally, each of the two treatments of utility uncertainty, regret-based and Bayesian, has its own advantages and drawbacks. We do not advocate for one over the other—each may be useful in different circumstances. Instead, we present a model and analysis of recommendation and query sets in both settings, and derive qualitatively similar results and observe related phenomenon in each. In particular, under both models of uncertainty, the relationship between optimal recommendation sets and optimal choice queries can be exploited effectively in the design of unified interactive recommendation algorithms that simultaneously accommodate both set-based preference elicitation and recommendation.

Acknowledgements. The authors would like to thank Darius Braziunas and Federico Toffano for discussions on minimax regret computation, Iain Murray and Cristina Manfredotti for helpful discussions on Monte Carlo methods, sampling techniques and particle filters and Nic Wilson for comments on a preliminary version of this article. This research was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada.

We thank the three anonymous reviewers for insightful comments and suggestions.

References

[1] Abbas, A. (2004). Entropy methods for adaptive utility elicitation. *IEEE Transactions on Systems, Science and Cybernetics*, 34(2):169–178.

[2] Adomavicius, G. and Kwon, Y. (2011). Maximizing aggregate recommendation diversity: a graph-theoretic approach. In *Proceedings of the Workshop on Novelty and Diversity in Recommender Systems (DiveRS-11) at the 5th ACM International Conference on Recommender Systems (RecSys-11)*, pages 3–10, Chicago, IL, USA.

[3] Ai, Q., Bi, K., Guo, J., and Croft, W. B. (2018). Learning a deep listwise context model for ranking refinement. In *Proceedings of the 41st Annual International ACM Conference on Research and Development in Information Retrieval (SIGIR-18)*, pages 135–144, Ann Arbor, MI, USA.

[4] Ailon, N., Karnin, Z. S., and Joachims, T. (2014). Reducing dueling bandits to cardinal bandits. In *Proceedings of the Thirty-first International Conference on Machine Learning (ICML-14)*, pages 856–864, Beijing, China.

[5] Akrour, R., Schoenauer, M., and Sebag, M. (2012). APRIL: active preference learning-based reinforcement learning. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML- PKDD 2012)*, pages 116–131, Bristol, UK.

[6] Baier, J. A. and McIlraith, S. A. (2008). Planning with preferences. *AI Magazine*, 29(4):25–36.

[7] Bana E Costa, C. A. and Vansnick, J.-C. (1994). MACBETH - an interactive path towards the construction of cardinal value functions. *International transactions in Operational Research*, 1:489–500.

[8] Bell, D. E. (1982). Regret in decision making under uncertainty. *Operations Research*, 30:961–981.

[9] Bello, I., Kulkarni, S., Jain, S., Boutilier, C., Chi, E., Eban, E., Luo, X., Mackey, A., and Meshi, O. (2018). Seq2slate: Re-ranking and slate optimization with RNNs. `arXiv:1810.02019`.

[10] Benabbou, N., Diodoro, S. D. S. D., Perny, P., and Viappiani, P. (2016). Incremental preference elicitation in multi-attribute domains for choice and ranking with the Borda count. In *Proceedings of the 10th International Conference on Scalable Uncertainty Management (SUM-16)*, pages 81–95, Nice, France.

[11] Benabbou, N. and Perny, P. (2015). Combining preference elicitation and search in multiobjective state-space graphs. In *Proceedings of the Twenty-fourth International Joint Conference on Artificial Intelligence (IJCAI-15)*, pages 297–303, Buenos Aires, Argentina.

[12] Benabbou, N., Perny, P., and Viappiani, P. (2014). Incremental elicitation of Choquet capacities for multicriteria decision making. In *Proceedings of the European Conference on Artificial Intelligence (ECAI-14)*, pages 87–92.

[13] Benabbou, N., Perny, P., and Viappiani, P. (2017). Incremental elicitation of Choquet capacities for multicriteria choice, ranking and sorting problems. *Artificial Intelligence*, 246:152–180.

[14] Berry, D. A. and Fristedt, B. (1985). *Bandit Problems: Sequential Allocation of Experiments*. Chapman and Hall, London.

[15] Bigot, D., Zanuttini, B., Fargier, H., and Mengin, J. (2013). Probabilistic conditional preference networks. In *Proceedings of the Twenty-ninth Conference on Uncertainty in Artificial Intelligence (UAI-13)*, Bellevue, WA, USA.

[16] Bonilla, E. V., Guo, S., and Sanner, S. (2010). Gaussian process preference elicitation. In *Advances in Neural Information Processing Systems 23 (NIPS-10)*, pages 262–270, Vancouver, BC, Canada.

[17] Bous, G. and Pirlot, M. (2013). Learning multicriteria utility functions with random utility models. In *Proceedings of the Third International Conference on Algorithmic Decision Theory (ADT-13)*, pages 101–115, Bruxelles, Belgium.

[18] Boutilier, C. (2002). A POMDP formulation of preference elicitation problems. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence (AAAI-02)*, pages 239–246, Edmonton, AB, Canada.

[19] Boutilier, C. (2011). Computational decision support: Regret-based models for optimization and preference elicitation. In Zentall, T. R. and Crowley, P. H., editors, *Comparative Decision Making: Analysis and Support Across Disciplines and Applications*, pages 423–453. Oxford University Press.

[20] Boutilier, C., Brafman, R., Domshlak, C., Hoos, H., and Poole, D. (2004a). Preference-based constrained optimization with CP-nets. *Computational Intelligence*, 20(2):137–157.

[21] Boutilier, C., Patrascu, R., Poupart, P., and Schuurmans, D. (2006). Constraint-based optimization and utility elicitation using the minimax decision criterion. *Artificial Intelligence*, 170(8–9):686–713.

[22] Boutilier, C., Sandholm, T., and Shields, R. (2004b). Eliciting bid taker non-price preferences in (combinatorial) auctions. In *Proceedings of the Nineteenth National Conference on Artificial Intelligence (AAAI-04)*, pages 204–211, San Jose, CA, USA.

[23] Boutilier, C., Zemel, R. S., and Marlin, B. (2003). Active collaborative filtering. In *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence (UAI-03)*, pages 98–106, Acapulco, Mexico.

[24] Braziunas, D. (2011). *Decision-theoretic Elicitation of Generalized Additive Utilities*. PhD thesis, University of Toronto.

[25] Braziunas, D. and Boutilier, C. (2007). Minimax regret based elicitation of generalized additive utilities. In *Proceedings of the Twenty-third Conference on Uncertainty in Artificial Intelligence (UAI-07)*, pages 25–32, Vancouver, BC, Canada.

[26] Braziunas, D. and Boutilier, C. (2008). Elicitation of factored utilities. *AI Magazine*, 29(4):79–92.

[27] Braziunas, D. and Boutilier, C. (2010). Assessing regret-based preference elicitation with the UTPREF recommendation system. In *Proceedings of the Eleventh ACM Conference on Electronic Commerce (EC'10)*, pages 219–228.

[28] Buchbinder, N., Feldman, M., Naor, J., and Schwartz, R. (2014). Submodular maximization with cardinality constraints. In *Proceedings of the Twenty-Fifth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA-14)*, pages 1433–1452, Portland, Oregon, USA.

[29] Burke, R. (2002). Interactive critiquing for catalog navigation in e-commerce. *Artificial Intelligence Review*, 18(3-4):245–267.

[30] Camerer, C. F., Loewenstein, G., and Rabin, M., editors (2003). *Advances in Behavioral Economics*. Princeton University Press, Princeton, New Jersey.

[31] Campos, P. G., Díez, F., and Cantador, I. (2014). Time-aware recommender systems: A comprehensive survey and analysis of existing evaluation protocols. *User Modeling and User-Adapted Interaction*, 24(1–2):67–119.

[32] Chajewska, U., Koller, D., and Parr, R. (2000). Making rational decisions using adaptive utility elicitation. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-00)*, pages 363–369, Austin, TX, USA.

[33] Chen, L. and Pu, P. (2006). Evaluating critiquing-based recommender agents. In *Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, Boston, MA, USA.

[34] Chen, L. and Pu, P. (2007). Hybrid critiquing-based recommender systems. In *Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI-07)*, pages 22–31, Honolulu, Hawaii, USA.

[35] Chen, L. and Pu, P. (2012). Critiquing-based recommenders: survey and emerging trends. *User Modeling and User-Adapted Interaction*, 22(1-2):125–150.

[36] Chen, M., Beutel, A., Covington, P., Jain, S., Belletti, F., and Chi, E. (2019). Top-k off-policy correction for a REINFORCE recommender system. In *Proceedings of the 12th ACM International Conference on Web Search and Data Mining (WSDM-19)*, pages 456–464, Melbourne, Australia.

[37] Cheng, H.-T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., and Shah, H. (2016). Wide & deep learning for recommender systems. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pages 7–10, Boston, MA, USA.

[38] Choi, S., Ha, H., Hwang, U., Kim, C., Ha, J.-W., and Yoon, S. (2018). Reinforcement learning-based recommender system using biclustering technique. `arXiv:1801.05532`.

[39] Chu, W. and Ghahramani, Z. (2005). Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:1019–1041.

[40] Corless, R. M., Gonnet, G. H., Hare, D. E., Jeffrey, D. J., and Knuth, D. E. (1996). On the Lambert W function. *Advances in Computational mathematics*, 5(1):329–359.

[41] Cornelio, C., Goldsmith, J., Mattei, N., Rossi, F., and Venable, K. B. (2013). Updates and uncertainty in CP-nets. In *Proceedings of the 26th Australasian Joint Conference (AI 2013)*, pages 301–312, Dunedin, New Zealand.

[42] Covington, P., Adams, J., and Sargin, E. (2016). Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys-16)*, pages 191–198, Boston, MA, USA.

[43] Craswell, N., Zoeter, O., Taylor, M., and Ramsey, B. (2008). An experimental comparison of click position-bias models. In *Proceedings of the 2008 International Conference on Web Search and Data Mining (WSDM-08)*, pages 87–94.

[44] desJardins, M., Eaton, E., and Wagstaff, K. (2006). Learning user preferences for sets of objects. In *Proceedings of the Twenty-third International Conference on Machine Learning (ICML-06)*, pages 273–280, Pittsburgh, USA.

[45] Dragone, P., Pellegrini, G., Vescovi, M., Tentori, K., and Passerini, A. (2018a). No more ready-made deals: Constructive recommendation for telco service bundling. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys-18)*, pages 163–171, Vancouver, British Columbia, Canada.

[46] Dragone, P., Teso, S., Kumar, M., and Passerini, A. (2018b). Decomposition strategies for constructive preference elicitation. In *Proceedings of the Thirty-second AAAI Conference on Artificial Intelligence (AAAI-18)*, pages 2934–2942, New Orleans, LA, USA.

[47] Ekstrand, M. D., Riedl, J., and Konstan, J. A. (2011). Collaborative filtering recommender systems. *Foundations and Trends in Human-Computer Interaction*, 4(2):175–243.

[48] Elahi, M., Ricci, F., and Rubens, N. (2013). Active learning strategies for rating elicitation in collaborative filtering: A system-wide perspective. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 5(1):13.

[49] Fishburn, P. C. (1967). Interdependence and additivity in multivariate, unidimensional expected utility theory. *International Economic Review*, 8:335–342.

[50] Gajos, K. and Weld, D. S. (2005). Preference elicitation for interface optimization. In *Proceedings of the 18th Annual ACM Symposium on User Interface Software and Technology (UIST-05)*, pages 173–182, Seattle, WA, USA.

[51] Gauci, J., Conti, E., Liang, Y., Virochsiri, K., He, Y., Kaden, Z., Narayanan, V., and Ye, X. (2018). Horizon: Facebook's open source applied reinforcement learning platform. `arXiv:1312.5602`.

[52] Gilbert, H., Spanjaard, O., Viappiani, P., and Weng, P. (2015). Reducing the number of queries in interactive value iteration. In *Proceedings of the Fourth International Conference on Algorithmic Decision Theory (ADT-15)*, pages 139–152, Lexington, KY, USA.

[53] Gomez-Uribe, C. A. and Hunt, N. (2016). The Netflix recommender system: Algorithms, business value, and innovation. *ACM Transactions on Management Information Systems*, 6(4):13:1–13:19.

[54] González, J., Dai, Z., Damianou, A. C., and Lawrence, N. D. (2017). Preferential Bayesian optimization. In *Proceedings of the Thirty-fourth international Conference on Machine Learning (ICML-17)*, pages 1282–1291, Sydney, Australia.

[55] Grabisch, M. (2016). *Set Functions, Games and Capacities in Decision Making*, volume 46 of *Theory and Decision Library C*. Springer International Publishing, The address.

[56] Grabisch, M. and Labreuche, C. (2010). A decade of application of the Choquet and Sugeno integrals in multi-criteria decision aid. *Annals of Operations Research*, 175(1):247–286.

[57] Grabisch, M., Marichal, J.-L., Mesiar, R., and Pap, E. (2009). *Aggregation Functions*. Encyclopedia of Mathematics and Its Applications. Cambridge University Press, New York, NY, USA, 1st edition.

[58] Greco, S., Mousseau, V., and Slowinski, R. (2008). Ordinal regression revisited: Multiple criteria ranking using a set of additive value functions. *European Journal of Operational Research*, 191:416–436.

[59] Guo, S. and Sanner, S. (2010). Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, (AISTATS-10)*, pages 289–296, Chia Laguna Resort, Sardinia, Italy.

[60] Guo, Y. and Gomes, C. P. (2009). Learning optimal subsets with implicit user preferences. In *Proceedings of the Twenty-first International Joint Conference on Artificial Intelligence (IJCAI-09)*, pages 1052–1057, Pasadena, CA, USA.

[61] Halpern, J. Y. (2005). *Reasoning about Uncertainty*. MIT Press.

[62] He, R. and McAuley, J. (2016). Fusing similarity models with Markov chains for sparse sequential recommendation. In *Proceedings of the IEEE International Conference on Data Mining (ICDM-16)*, Barcelona, Spain.

[63] Herbrich, R., Minka, T., and Graepel, T. (2006). Trueskill$^{\text{tm}}$: A Bayesian skill rating system. In *Advances in Neural Information Processing Systems 19 (NIPS-06)*, pages 569–576, Vancouver, BC, Canada.

[64] Herlocker, J. L., Konstan, J. A., Borchers, A., and Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR-99)*, pages 230–237, Berkeley, CA, USA.

[65] Hines, G. and Larson, K. (2010). Preference elicitation for risky prospects. In *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS.10)*, pages 889–896, Toronto, ON, Canada.

[66] Holloway, H. A. and White, III, C. C. (2003). Question selection for multiattribute decision-aiding. *European Journal of Operational Research*, 148:525–543.

[67] Howard, R. A. and Matheson, J. E., editors (1984). *Readings on the Principles and Applications of Decision Analysis*. Strategic Decision Group, Menlo Park, CA.

[68] Ie, E., Jain, V., Wang, J., Navrekar, S., Agarwal, R., Wu, R., Cheng, H.-T., Chandra, T., and Boutilier, C. (2019). SlateQ: A tractable decomposition for reinforcement learning with recommendation sets. In *Proceedings of the Twenty-eighth International Joint Conference on Artificial Intelligence (IJCAI-19)*, pages 2592–2599, Macau,China.

[69] Iyengar, V. S., Lee, J., and Campbell, M. (2001). Q-Eval: Evaluating multiple attribute items using queries. In *Proceedings of the Third ACM Conference on Electronic Commerce (EC'01)*, pages 144–153, Tampa, FL, USA.

[70] Jacobson, K., Murali, V., Newett, E., Whitman, B., and Yon, R. (2016). Music personalization at Spotify. In *Proceedings of the 10th ACM Conference on Recommender Systems (RecSys-16)*, pages 373–373, Boston, Massachusetts, USA.

[71] Jacquet-Lagrèze, E. and Siskos, Y. (1982). Assessing a set of additive utility functions for multicriteria decision making: the UTA method. *European Journal of Operational Research*, 10:151–164.

[72] Jiang, R., Gowal, S., Mann, T. A., and Rezende, D. J. (2019). Beyond greedy ranking: Slate optimization via list-CVAE. In *Proceedings of the Seventh International Conference on Learning Representations (ICLR-19)*, New Orleans, LA, USA.

[73] Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-02)*, pages 133–142, Edmonton, AB, Canada.

[74] Keeney, R. L. and Raiffa, H. (1976). *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley and Sons, New York.

[75] Kohli, P., Salek, M., and Stoddard, G. (2013). A fast bandit algorithm for recommendation to users with heterogenous tastes. In *Proceedings of the Twenty-seventh AAAI Conference on Artificial Intelligence (AAAI-13)*, pages 1135–1141, Bellevue, WA, USA.

[76] Konstan, J. A., Miller, B. N., Maltz, D., Herlocker, J. L., Gordon, L. R., and Riedl, J. (1997). Grouplens: Applying collaborative filtering to usenet news. *Communications of the ACM*, 40(3):77–87.

[77] Koren, Y. and Bell, R. M. (2015). Advances in collaborative filtering. In Ricci, F., Rokach, L., and Shapira, B., editors, *Recommender Systems Handbook*, pages 77–118. Springer.

[78] Kouvelis, P. and Yu, G. (1997). *Robust Discrete Optimization and Its Applications*. Kluwer, Dordrecht.

[79] Kunaver, M. and Pozrl, T. (2017). Diversity in recommender systems - A survey. *Knowledge-Based Systems*, 123:154–162.

[80] Kveton, B., Szepesvari, C., Wen, Z., and Ashkan, A. (2015). Cascading bandits: Learning to rank in the cascade model. In *Proceedings of the Thirty-second International Conference on Machine Learning (ICML-15)*, pages 767–776, Lille, France.

[81] Labreuche, C. and Huédé, F. L. (2005). Miriad: a tool suite for mcda. In *Proceedings of EUSFLAT'05*, pages 204–209.

[82] Leskovec, J., Krause, A., Guestrin, C., Faloutsos, C., VanBriesen, J. M., and Glance, N. S. (2007). Cost-effective outbreak detection in networks. In *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-07)*, pages 420–429, San Jose, California, USA.

[83] Loepp, B., Hussein, T., and Ziegler, J. (2014). Choice-based preference elicitation for collaborative filtering recommender systems. In *Proceedings of the 2014 SIGCHI Conference on Human Factors in Computing Systems (CHI-14)*, pages 3085–3094, Toronto, ON, Canada.

[84] Loomes, G. and Sugden, R. (1982). Regret theory: An alternative theory of rational choice under uncertainty. *Economic Journal*, 92:805–824.

[85] Louviere, J. J., Hensher, D. A., and Swait, J. D. (2000). *Stated Choice Methods: Analysis and Application*. Cambridge University Press, Cambridge.

[86] Lu, T. and Boutilier, C. (2011). Robust approximation and incremental elicitation in voting protocols. In *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI-11)*, pages 287–293, Barcelona, Catalonia, Spain.

[87] Lucas, C. G., Griffiths, T. L., Xu, F., and Fawcett, C. (2008). A rational model of preference learning and choice prediction by children. In

Advances in Neural Information Processing Systems 21 (NIPS-08), pages 985–992.

[88] Luce, R. D. (1959). *Individual choice behavior: A theoretical analysis*. Wiley.

[89] Marichal, J.-L., Meyer, P., and Roubens, M. (2005). Sorting multi-attribute alternatives: the TOMASO method. *Computers & Operations Research*, 32:861–877.

[90] McFadden, D. (1974). Conditional logit analysis of qualitative choice behavior. In Zarembka, P., editor, *Frontiers in Econometrics*, pages 105–142. Academic Press.

[91] McGinty, L. and Reilly, J. (2011). On the evolution of critiquing recommenders. In Ricci, F., Rokach, L., Shapira, B., and Kantor, P. B., editors, *Recommender Systems Handbook*, pages 419–453. Springer.

[92] Minka, T. P. (2001). Expectation propagation for approximate Bayesian inference. In *Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence (UAI-01)*, pages 362–369, Seattle, WA, USA.

[93] Minoux, M. (1978). Accelerated greedy algorithms for maximizing submodular set functions. In *Proceedings of the 8th IFIP Conference on Optimization Techniques*, pages 234–243, Würzburg, Germany.

[94] Naamani-Dery, L., Golan, I., Kalech, M., and Rokach, L. (2015). Preference elicitation for group decisions using the Borda voting rule. *Group Decision and Negotiation*, 24(6):1015–1033.

[95] Neal, R. M. (2003). Slice sampling. *The Annals of Statistics*, 31(3):705–70.

[96] Nemhauser, G. L., Wolsey, L. A., and Fisher, M. L. (1978). An analysis of approximations for maximizing submodular set functions—i. *Mathematical Programming*, 14(1):265–294.

[97] Ng, A. and Russell, S. (2000). Algorithms for inverse reinforcement learning. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-00)*, pages 663–670, Stanford, CA, USA.

[98] Perny, P., Viappiani, P., and Boukhatem, A. (2016). Incremental preference elicitation for decision making under risk with the rank-dependent utility model. In *Proceedings of the Thirty-second Conference on Uncertainty in Artificial Intelligence (UAI-16)*, New York City, NY, USA.

[99] Perrault, A. and Boutilier, C. (2019). Experiential preference elicitation for autonomous heating and cooling systems. In *Proceedings of the Eighteenth Conference on Autonomous Agents and Multiagent Systems (AAMAS-19)*, pages 431–439, Montreal, QC, Canada.

[100] Price, R. and Messinger, P. R. (2005). Optimal recommendation sets: Covering uncertainty over user preferences. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)*, pages 541–548, Pittsburgh, PA, USA.

[101] Regan, K. and Boutilier, C. (2009). Regret-based reward elicitation for Markov decision processes. In *Proceedings of the Twenty-fifth Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 454–451, Montreal, QC, Canada.

[102] Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. (2004). Dynamic critiquing. In *Proceedings of the 7th European Conference on Advances in Case-Based Reasoning (ECCBR-04)*, pages 763–777, Madrid, Spain.

[103] Reilly, J., McCarthy, K., McGinty, L., and Smyth, B. (2005). Incremental critiquing. *Knowledge-Based Systems*, 18(4–5):143–151.

[104] Reilly, J., Zhang, J., McGinty, L., Pu, P., and Smyth, B. (2007). Evaluating compound critiquing recommenders: a real-user study. In *Proceedings of the Eighth ACM Conference on Electronic Commerce (EC'07)*, pages 114–123, San Diego, California, USA.

[105] Rendle, S., Freudenthaler, C., and Schmidt-Thieme, L. (2010). Factorizing personalized Markov chains for next-basket recommendation. In *Proceedings of the 19th International World Wide Web Conference (WWW-10)*, pages 811–820, Raleigh, NC, USA.

[106] Rennie, J. and Srebro, N. (2005). Fast maximum margin matrix factorization for collaborative prediction. In *Proceedings of the Twenty-second International Conference on Machine Learning (ICML-05)*, Bonn, Germany.

[107] Riquelme, C., Tucker, G., and Snoek, J. (2018). Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for thompson sampling. In *Proceedings of the Sixth International Conference on Learning Representations (ICLR-18)*, Vancouver, BC, USA.

[108] Saaty, T. (1980). *The Analytic Hierarchy Process, Planning, Piority Setting, Resource Allocation*. McGraw-Hill, New york.

[109] Sahoo, N., Singh, P. V., and Mukhopadhyay, T. (2012). A hidden Markov model for collaborative filtering. *Management Information Systems Quarterly*, 36(4).

[110] Salakhutdinov, R. and Mnih, A. (2007). Probabilistic matrix factorization. In *Advances in Neural Information Processing Systems 20 (NIPS-07)*, pages 1257–1264, Vancouver, BC, Canada.

[111] Salo, A. and Hämäläinen, R. P. (2001). Preference ratios in multiattribute evaluation (PRIME)–elicitation and decision procedures under incomplete information. *IEEE Transactions on Systems, Man, and Cybernetics*, 31(6):533–545.

[112] Salo, A. and Hämäläinen, R. P. (2010). Preference programming multicriteria weighting models under incomplete information. In *Handbook of Multicriteria Analysis*, volume 103 of *Applied Optimization*, pages 167–187. Springer.

[113] Savage, L. J. (1951). The theory of statistical decision. *Journal of the American Statistical Association*, 46(253):55–67.

[114] Savage, L. J. (1954). *The Foundations of Statistics*. Wiley, New York.

[115] Selvin, S., Bloxham, M., Khuri, A. I., Moore, M., Coleman, R., Bryce, G. R., Hagans, J. A., Chalmers, T. C., Maxwell, E. A., and Smith, G. N. (1975). Letters to the editor. *The American Statistician*, 29(1):pp. 67–71.

[116] Shani, G., Heckerman, D., and Brafman, R. I. (2005). An MDP-based recommender system. *Journal of Machine Learning Research*, 6:1265–1295.

[117] Sui, Y., Zhuang, V., Burdick, J. W., and Yue, Y. (2017). Multi-dueling bandits with dependent arms. In *Proceedings of the Thirty-third Conference on Uncertainty in Artificial Intelligence (UAI-17)*, Sydney, Australia.

[118] Swaminathan, A., Krishnamurthy, A., Agarwal, A., Dudik, M., Langford, J., Jose, D., and Zitouni, I. (2017). Off-policy evaluation for slate recommendation. In *Advances in Neural Information Processing Systems 30 (NIPS-17)*, pages 3632–3642, Long Beach, CA, USA.

[119] Taghipour, N., Kardan, A., and Ghidary, S. S. (2007). Usage-based web recommendations: A reinforcement learning approach. In *Proceedings of the First ACM Conference on Recommender Systems (RecSys07)*, pages 113–120, Minneapolis, MN, USA. ACM.

[120] Tan, Y. K., Xu, X., and Liu, Y. (2016). Improved recurrent neural networks for session-based recommendations. In *Proceedings of the 1st Workshop on Deep Learning for Recommender Systems*, pages 17–22, Boston, MA, USA.

[121] Tehrani, A. F., Cheng, W., Dembczynski, K., and Hüllermeier, E. (2011). Learning monotone nonlinear models using the Choquet integral. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML- PKDD 2011)*, pages 414–429, Athens, Greece.

[122] Teso, S., Dragone, P., and Passerini, A. (2017a). Coactive critiquing: Elicitation of preferences and features. In *Proceedings of the Thirty-first*

AAAI Conference on Artificial Intelligence (AAAI-17), pages 2639–2645, San Francisco, California, USA.

[123] Teso, S., Passerini, A., and Viappiani, P. (2016). Constructive preference elicitation by setwise max-margin learning. In *Proceedings of the Twenty-fifth International Joint Conference on Artificial Intelligence (IJCAI-16)*, pages 2067–2073, New York, NY, USA.

[124] Teso, S., Passerini, A., and Viappiani, P. (2017b). Constructive preference elicitation for multiple users with setwise max-margin. In *Proceedings of the Fifth International Conference on Algorithmic Decision Theory (ADT-17)*, pages 3–17, Luxembourg, Luxembourg.

[125] Torra, V. and Narukawa, Y. (2007). *Modeling Decisions - Information Fusion and Aggregation Operators*. Springer.

[126] Toubia, O., Hauser, J. R., and Simester, D. I. (2004). Polyhedral methods for adaptive choice-based conjoint analysis. *Journal of Marketing Research*, 41(1):116–131.

[127] Train, K. E. (2009). *Discrete Choice Methods with Simulation*. Cambridge University Press, Cambridge, United Kingdom.

[128] Tversky, A. and Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science*, 185(4157):1124–1131.

[129] van den Oord, A., Dieleman, S., and Schrauwen, B. (2013). Deep content-based music recommendation. In *Advances in Neural Information Processing Systems 26 (NIPS-13)*, pages 2643–2651, Lake Tahoe, NV, USA.

[130] Vargas, S., Baltrunas, L., Karatzoglou, A., and Castells, P. (2014). Coverage, redundancy and size-awareness in genre diversity for recommender systems. In *Proceedings of the Eighth ACM Conference on Recommender Systems (RecSys-14)*, pages 209–216, Foster City, Silicon Valley, CA, USA.

[131] Viappiani, P. and Boutilier, C. (2009). Regret-based optimal recommendation sets in conversational recommender systems. In *Proceedings of the 3rd ACM Conference on Recommender Systems (RecSys-09)*, pages 101–108, New York, NY, USA.

[132] Viappiani, P. and Boutilier, C. (2010). Optimal bayesian recommendation sets and myopically optimal choice query sets. In *Advances in Neural Information Processing Systems 23 (NIPS-10)*, pages 2352–2360, Vancouver, BC, Canada.

[133] Viappiani, P. and Boutilier, C. (2011). Recommendation sets and choice queries: There is no exploration/exploitation tradeoff! In *Proceedings of the Twenty-fifth AAAI Conference on Artificial Intelligence (AAAI-11)*, pages 1571–1574, San Francisco, CA, USA.

[134] Viappiani, P., Faltings, B., and Pu, P. (2006). Preference-based search using example-critiquing with suggestions. *Journal of Artificial Intelligence Research*, 27:465–503.

[135] Viappiani, P. and Kroer, C. (2013). Robust optimization of recommendation sets with the maximin utility criterion. In *Proceedings of the Third International Conference on Algorithmic Decision Theory (ADT-13)*, pages 411–424.

[136] Wald, A. (1950). *Statistical Decision Functions*. Wiley, New York.

[137] Wang, H., Wang, N., and Yeung, D.-Y. (2015). Collaborative deep learning for recommender systems. In *Proceedings of the Twenty-first ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-15)*, pages 1235–1244, Sydney, Australia.

[138] Wang, Y., Ouyang, H., Wang, C., Chen, J., Asamov, T., and Chang, Y. (2017). Efficient ordered combinatorial semi-bandits for whole-page recommendation. In *Proceedings of the Thirty-first AAAI Conference on Artificial Intelligence (AAAI-17)*, pages 2746–2753, San Francisco, CA, USA.

[139] Weng, P. and Zanuttini, B. (2013). Interactive value iteration for Markov decision processes with unknown rewards. In *Proceedings of the Twenty-third International Joint Conference on Artificial Intelligence (IJCAI-13)*, pages 2415–2421, Beijing, China.

[140] Wilhelm, M., Ramanathan, A., Bonomo, A., Jain, S., Chi, E. H., and Gillenwater, J. (2018). Practical diversified recommendations on YouTube with determinantal point processes. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management (CIKM-18)*, pages 2165–2173, Torino, Italy.

[141] Wu, C.-Y., Ahmed, A., Beutel, A., Smola, A. J., and Jing, H. (2017). Recurrent recommender networks. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining (WSDM-17)*, pages 495–503, Cambridge, UK.

[142] Yager, R. R. (1998). On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transactions on Systems, Man, and Cybernetics*, 18:183–190.

[143] Zhang, J. and Pu, P. (2006). A comparative study of compound critique generation in conversational recommender systems. In *Proceedings of the 4th International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems (AH-06)*, pages 234–243, Dublin, Ireland.

[144] Zhao, X., Xia, L., Zhang, L., Ding, Z., Yin, D., and Tang, J. (2018). Deep reinforcement learning for page-wise recommendations. In *Proceedings of the 12th ACM Conference on Recommender Systems (RecSys-18)*, pages 95–103, Vancouver, BC, Canada.

[145] Ziegler, C., McNee, S. M., Konstan, J. A., and Lausen, G. (2005). Improving recommendation lists through topic diversification. In *Proceedings of the 14th international conference on World Wide Web, (WWW-05)*, pages 22–32, Chiba, Japan.

Appendix A. Monty Hall Problem

Consider the following *3-item recommender problem* where we have items $X = \{a, b, c\}$. The recommender systems is allowed to ask *a single* comparison query to the user, mister Monty, and then needs to provide a final recommendation. Let's further assume that utility values $(u(a), u(b), u(c))$ belong to $W = \{(1, 0, 0), (0, 1, 0), (0, 0, 1)\}$, all realized with equal probability prior $\frac{1}{3}$, that constitute the initial belief θ^0. The value of the set $S = \{b, c\}$ as recommendation set in this initial belief θ^0 is

$$\mathrm{EUS}(S;\theta^0) = \sum_{w \in W} \max\left[u(b), u(c)\right] \cdot P(w;\theta^0) = 0.33 \cdot \max(0,0) + 0.33 \cdot \max(0,1) + 0.33 \cdot \max(1,0) = 0.66. \quad \text{(A.1)}$$

The set S is an optimal recommendation set of size $k = 2$, as no other pair can be better, given the symmetric and "egalitarian" prior. Then, in virtue of our theorem for noiseless responses (Theorem 25), an optimal recommendation set is also an optimal query set:

$$\mathrm{EPU}(S\,;\theta^0) = \mathrm{EPU}^*, \text{ and } \mathrm{EPU}(S\,;\theta^0) = \mathrm{EUS}(S\,;\theta^0) = 0.66.$$

In fact any set $\{a, b\}$, $\{a, c\}$ and $\{b, c\}$ is optimal in θ^0; let's assume to pick $\{b, c\}$. We then ask Mr Monty to identify, among b and c, the item that he prefers. Upon receiving the answer, we now have to make a (single) recommendation; should we recommend item a, b, or c ?

In fact, we know that the optimal recommendation/query set is a fixed point for operator T (Corollary 26). This means, that the optimal recommendation in the updated belief $\theta^0|S \rhd b$ coincides with the user's selection from the set (Proposition 21). As the query is between item b and item c, the system should recommend item b or item c depending on which is selected/preferred by Monty (the item a is never optimal in expected utility in any of the two cases). Assume, with no loss of generality, that the answer is b; then we should recommend this item to him.

If we want to calculate precisely the expected utility of recommending an item, we can proceed as follows. We define beliefs $\theta^1 := \theta^0|S \rhd b$ (item b is preferred) and $\theta^2 := \theta^0|S \rhd c$ (item c is preferred). From the equivalence $\mathrm{EUS}(S\,;\theta^0) = \mathrm{EPU}(S\,;\theta^0) = 0.66$ we derive (substituing the definition of EPU)

$$\mathrm{EUS}(S\,;\theta^0) = P(S \rhd b; \theta^0)EU(b; \theta^1) + P(S \rhd c; \theta^0)EU(c; \theta^2) \text{ and } P(S \rhd b; \theta^0) = P(S \rhd c; \theta^0) = 0.5$$

and then it has to be $\mathrm{EU}(b, \theta^1) = \mathrm{EU}(c; \theta^2) = 0.66$. Because Proposition 21, b is optimal in θ^1 (respectively c is optimal in θ^2) and expected utility in the posterior belief is 0.66.

This problem is formally equivalent to the famous Monty Hall problem [115]:

Suppose you're on a game show, and you're given the choice of three doors: behind one door is a car; behind the others, goats. You pick a door, say No. 1, and the host, who knows what's behind the doors, opens another door, say No. 3, which has a goat. He then says to you, "Do you want to pick door No. 2?" Is it to your advantage to switch your choice?

We make the further assumption of an explicit uniform prior on possible locations of the car and a sound *response model* for Monty; these assumptions are not explicitly stated in the original problem, but taken for granted in the solution (see [61], pag. 216, for a discussion of the implications of the possible assumptions on Monty's "response model"; i.e. protocols in Halpern's formalization).

After you initially pick a door, the fact that Monty opens a particular door, revealing a goat, is a signal for the unopened door to be likely to have the prize. Assume you pick door number 1; Monty opens a door according to the following "response model": if both door no. 2 and 3 have a goat, he opens one of the two doors at random; otherwise he opens the only door with the goat between no. 2 and door no. 3.

The analogy between the 3-item recommendation problem above and the Monty Hall puzzle is completed with the following correspondences: items a, b and c corresponds to, respectively doors 1, 2 and 3; goats correspond to items with utility 0, the car corresponds to utility 1 and the action of picking the initial door corresponds to a *choice query* between the *two other doors/items* with Monty answering according to the response model R_{NL}. In fact, when Monty reveals a goat behind door no. 3, it means that door no. 2 is "preferred" to the other, door no. 1 (while the precise numerical value of the "utility" of neither item/door is revealed).

The strategy "keep-same-door" in the Monty hall problem corresponds to the recommendation of the item not included in the choice query in our "recommender problem" with 3 items; while the strategy "switch-door" in the original puzzle corresponds to picking the item that is preferred by Monty in the choice query rather than the item a, that was not part of the query (no rational strategy will recommend the item c that is stated to be least preferred than b; assuming Monty's answers to be noiseless). As discussed before, since the comparison query is an optimal query, it is a fixed point for the T operator (Corollary 26), and the optimal recommendation in the updated belief must necessarily agree with Monty's response (Proposition 21). Our reasoning shows that "switch-door" is the best action, and it gives (according to Equation A.1) a probability of 0.66 of picking the right door/item.

Using our framework for evaluating queries and sets, we are able to solve the Monty Hall puzzle in what we consider a very elegant form without an explicit Bayesian update.

Appendix B. Bound for the Logistic Noise Model

Theorem 30 *Assume a set selection problem with known user utility, in which the user with utility w—inducing item utilities $v_1, \dots, v_k$—selects an item from slate S (of size k). The expected loss induced by user choice under response model R_L is at most:*

$$\hat{\Delta}^k = \frac{1}{\gamma} \cdot \mathcal{L}(\frac{k-1}{e}), \tag{76}$$

where $\mathcal{L}$ is the Lambert W function [40]. Moreover, the maximum loss is attained by utilities $v_1, \dots, v_k$:

$$v_1 = \ldots = v_{k-1} = \zeta \tag{77}$$

$$v_k \qquad = \zeta + \frac{1}{\gamma} \cdot \left[1 + \mathcal{L}\big(\frac{k-1}{e}\big)\right] \tag{78}$$

with $\zeta \in \mathbb{R}$, or any permutation of these values (since the loss does not depend on the ordering of the values $v_1, \dots, v_k$).

Proof. *Case $k=2$ (pairs of items)* We consider a two items x_1 and x_2. Let $z = u(x_1) - u(x_2)$ be the difference in utility; assume $z \geq 0$ with no loss of generality. The expected loss is exactly the utility difference z times the probability of choosing the less preferred option under R_L: $1 - L(\gamma z) = L(-\gamma z)$ where L is the logistic function.

$$\delta(z) = \frac{z}{1 + e^{\gamma z}} \tag{B.1}$$

We are looking for $\hat{\delta} = \max \delta(z)$. The derivative of the loss $\delta(z)$ wrt z is

$$\frac{d\delta(z)}{dz} = \frac{1}{1 + e^{\gamma z}} + \gamma z \frac{-e^{\gamma z}}{(1 + e^{\gamma z})^2} = \frac{1}{1 + e^{\gamma z}}\left(1 - \gamma z \frac{e^{\gamma z}}{1 + e^{\gamma z}}\right). \tag{B.2}$$

We impose the derivative equal to zero ah solve the equation in z in order to find $\hat{z}$:

$$\frac{d\delta(z)}{dz} = 0 \;\Leftrightarrow\; 1 + e^{\gamma z} - \gamma z e^{\gamma z} = 0 \;\Leftrightarrow\; (\gamma \hat{z} - 1)\, e^{\gamma \hat{z} - 1} = e^{-1} \;\Leftrightarrow\; \gamma \hat{z} - 1 = \mathcal{L}\left(\frac{1}{e}\right) \;\Leftrightarrow\; \hat{z} = \frac{1}{\gamma}\left[1 + \mathcal{L}\left(\frac{1}{e}\right)\right] \tag{B.3}$$

where $\mathcal{L}(\cdot)$ is the Lambert-W function. The last expression of Eq. B.2, substituted into Eq. B.1, gives $\hat{\delta} = \delta(\hat{z}) = \frac{e^{-\gamma \hat{z}}}{\gamma} = \hat{z} - \frac{1}{\gamma}$. Thus:

$$\hat{z} = \frac{1}{\gamma}\left[1 + \mathcal{L}\left(\frac{1}{e}\right)\right] \quad ; \qquad \hat{\delta} = \frac{1}{\gamma}\mathcal{L}\left(\frac{1}{e}\right). \tag{B.4}$$

The second derivative is

$$\left.\frac{d^2\delta(z)}{d^2 z}\right|_{z=\hat{z}} = -\frac{1}{\hat{z}\, e^{\gamma \hat{z}}} < 0$$

confirming that $\hat{z}$ is a (relative) maximum; by the fact that $\hat{z}$ is the only real value with null derivative, and that $\delta(0) = 0 < \hat{\delta}$, we conclude that $\hat{z}$ is an absolute maximum.

General case ($k \geq 2$) We consider the loss when choosing an item among k possibilities. Let's assume that $v_1, \dots, v_k$ are the utility values attained by the k items. The logistic response model is such that $P_L(S \rightsquigarrow x_i) = \frac{e^{v_i}}{\sum_{j=1}^k e^{v_j}}$.

Assume, with no loss of generality, that $v_1 \leq \ldots \leq v_k$. In order to determine the bound, we need to maximize the following function:

$$(v_k - v_1)\frac{e^{v_1}}{e^{v_1} + \ldots + e^{v_k}} + \ldots + (v_k - v_{k-1})\frac{e^{v_{k-1}}}{e^{v_1} + \ldots + e^{v_k}} = \sum_{i=1}^{k}(v_k - v_i)\frac{e^{v_i}}{\sum_{j=1}^k e^{v_j}} = \tag{B.5}$$

$$= \sum_{i=1}^{k-1}(v_k - v_i)\frac{1}{1 + \sum_{j \neq i} e^{v_j - v_i}}. \tag{B.6}$$

We notice that the expected loss is expressed as a function of terms $v_j - v_i$. We set $z_i = v_k - v_i$, equal to the utility difference between v_k and v_i (that is ≥ 0, given the assumption that v_k is the highest value); in this way, we are left with a problem with $k-1$ free variables, representing the utility loss of each item with respect to the kth item.

$$f(z_1, \ldots, z_{k-1}) = \sum_{i \in K^-} z_i \frac{1}{1 + e^{z_i} + \sum_{j \neq i} e^{z_i - z_j}} \tag{B.7}$$

where $K^- = \{1, \ldots, k-1\}$.

We look for null gradient by setting the partial derivatives to 0:

$$\frac{\partial f}{\partial z_i} = 0 \iff e^{\sum_j z_j}(z_i - 1) + \sum_{j \neq i} \left[e^{\sum_{l \neq j} z_l}(z_i - z_j - 1) \right] - e^{\sum_{j \neq i} z_j} = 0 \tag{B.8}$$

where all indices i, j, l are meant to belong to K^-. By subtracting the expression B.8 for a index j from the expression of another index i, we obtain:

$$\frac{\partial f}{\partial z_i} - \frac{\partial f}{\partial z_j} = 0 \iff \left[e^{\sum_l z_l} + \sum_l e^{\sum_{o \neq l} z_o} \right](z_i - z_j) = 0 \iff z_i = z_j. \tag{B.9}$$

Therefore we conclude that it must be $z_1 = z_2 = \ldots = z_{k-1}$ in order for the gradient to be null. We therefore restrict our attention to solutions such that all z_i are the same:

$$f(z, \ldots, z) = \frac{(k-1)z}{k - 1 + e^{\gamma z}}. \tag{B.10}$$

First order conditions require that:

$$k - 1 + e^{\gamma z} - \gamma z e^{\gamma z} = 0. \tag{B.11}$$

We solve the equation in z in order to find $\hat{z}$:

$$(\gamma \hat{z} - 1)e^{\gamma \hat{z}} = k - 1 \ \Leftrightarrow (\gamma \hat{z} - 1)\, e^{\gamma \hat{z} - 1} = (k-1)e^{-1} \ \Leftrightarrow \gamma \hat{z} - 1 = \mathcal{L}\left(\frac{k-1}{e}\right) \Leftrightarrow \hat{z} = \frac{1}{\gamma}\left[1 + \mathcal{L}\left(\frac{k-1}{e}\right)\right] \tag{B.12}$$

where $\mathcal{L}(\cdot)$ is the Lambert-W function.

$$\hat{\Delta} = f(\hat{z}) = (k-1)\frac{e^{-\gamma \hat{z}}}{\gamma} = \hat{z} - \frac{1}{\gamma} \tag{B.13}$$

We finally obtain:

$$\hat{z} = \frac{1}{\gamma}\left[1 + \mathcal{L}\left(\frac{k-1}{e}\right)\right] \quad ; \qquad \hat{\Delta} = \frac{1}{\gamma}\mathcal{L}\left(\frac{k-1}{e}\right). \tag{B.14}$$

□